Mikrotik - Part 7

User Topics

PDF generated using the open source mwlib toolkit. See http://code.pediapress.com/ for more information.
PDF generated at: Thu, 19 Dec 2013 20:05:50 CET

Contents

Articles

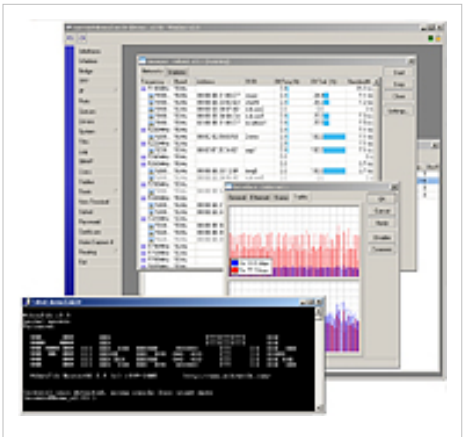
MikroTik RouterOS 1
Hardware 2
Supported Hardware 2
Bandwidth Managment and Queues 24
Firewall 25
Monitoring 26
User/Routing 27
Scripts 29
Tunnels 32
Wireless Setups 32
Manual:MPLS 33
Manual:Virtualization 36
Use Metarouter to Implement Tor Anonymity Software 37
User/IPv6 43
User Management 43
The Dude 44
User Manager 45
API PHP package 48
API in C using winsock 57
Manual:API Python3 77
API multiplatform from townet 81
MikroNode 86
API in Java 91
API In CPP 104
Api php template 132
API in VB dot NET 156
RouterOS PHP class 160
API command notes 176
API Ruby class 186
Librouteros 202
API in C 204
API ActionScript 3 class 239
API Delphi Client 246
API Delphi 249

API in C Sharp 252
API PHP class 258
MikroTik for Mac 271
Assorted examples 274

References

Article Sources and Contributors 275
Image Sources, Licenses and Contributors 277

MikroTik RouterOS



This is a user editable how-to page, anyone can contribute! If you have some firewall rules or scripts to share, simply register and add to the list. This is not the official Mikrotik Manual, this page is maintained by our users - by You. Anyone can join and share their configuration, setups, ideas and manuals. And if you find a mistake in someone elses article - simply correct it.

Topics

- Hardware
  - Supported Hardware
- Bandwidth Control
- Firewall
- Monitoring
- Routing
- Scripts
- VPN
- Wireless
- MPLS
- Virtualization
  - Virtualization manual
  - Use Metarouter to Implement Tor Anonymity Software
- IPv6

Management and Monitoring

- User management, Hotspots
- The Dude
- User Manager

Miscellaneous Stuff

- MikroTik for Mac
- Assorted examples

Translations

- Articles in Serbian
- Articles in Turkish
- Articles in Spanish
- Articles in Bulgarian
- Articles in Croatian
- Articles in Portugese
- Articles in Russian

Hardware

- RB 750G - Getting Started Help

Supported Hardware

This page should be edited by the user community to reflect their tested hardware and version used.

See also: Device driver list [1] in manual

Motherboards

Vendor	Model	ROS version	Result
Asrock	**Intel 82801G chipset**	3.0-3.14	Bad performance, locks up under heavy load, supports multi cpu, PATA not supported, integrated ethernet not recognized. Maybe it's just Asrock bad motherboard don't know if the problem is in intel 82801G chipset tested on 2 motherboards, never tested on 2.9.x
ASUS	**P8H67-V (Intel® H67(B3), 3xPCI, 4xPCI-E)**	5.XX	Don't work. "Kernel Panic". Mikrotik has not driver compatibility with this motherboard.
ASUS	**P5B-Deluxe (Intel P965, 3xPCI, 3xPCI-E)**	3.13	Works fine on Intel Core 2 Duo E6400. Not supported PATA controller. Both integrated Gigabit NIC (Marvell Yukon 88E8056 & 88E8001) works fine but only at 100Mbps.
ASUS	**P5B-V (Intel P965, 3xPCI, 4xPCI-E)**	3.13	Works fine on Intel Dual Core E2180. Not supported PATA controller. Integrated Gigabit NIC Marvell Yukon 88E8001 works fine but only at 100Mbps. Winbox via MAC = problem, disconnects after 3 seconds. Winbox via IP no problem.
ASUS	**P5KC (Intel P35, 3x PCI, 3xPCI-E)**	3.10	Not supported PATA controller (JMicron JMB363), ROS can boot from USB flash drive; internal ethernet not recognized.
ASUS	**P5GC-MX/1333 (2x PCI)**	3.7	Works great for pentium dual-core e2160, hdd pata and sata, 1,5gb ram dual channel mode, except the attansic l2 ethernet onboard card is not recognized.
ASUS	**P5GC (6xPCI)**	2.9.39	Ethernet recognized but not working
ASUS	**P7P55D PRO**	4.5	Works ok, PATA controller (JMicron JM363) and internal Ethernet successfully recognized.
ASUS	**P6T SE**	4.0	RouterOS boots and works with SATA disk set to 'IDE compatibility mode'.
ASUS	**A7V133-C**	3.0 beta 7	Works fine
ASUS	**A7V600-X**	3.25	Works fine
EPoX	**EP-4VKMI**	3.16 - 3.24	Works fine.
EPoX	**8RDA+**	3.6	Work fine including integrated ethernet
Intel	**D815EGEW**	2.9.x	Excellent performance under 2.9. Not tested under v3. Onboard Ethernet Works perfectly.
Intel	**D845GVSRL**	3.7, 3.1, 2.9	Very Stable, used for 4 years
Intel	**D945GCCRL**	2.9.43 & 3.0 beta 5	Ethernet & DoM not recognized
Intel	**D945GCLF2**	3.23	4 core's, 2gb ram, 32gb ssd, no problems.
Intel	**D945GCPE**	3.0 beta 9	Work fine including integrated ethernet

Intel	**D945GCNL**	3.11	Works fine but integrated ethernet (just disable) goes up and down on reboots multi-cpu= yes. shared IRQ for PCI devices, decrease nic performace.
Intel	**D945GNT**	2.9.45	Works fine
Intel	**DG33FB**	3.7	Works fine, Ethernet but not working (IRQ 9), set in BIOS Security/XD Tegnology to disable
Intel	**DG950**	2.9.42	Ethernet not recognized
Intel	**DG965SWH**	3.0 beta 9	Works fine, but only with SATA not IDE
Intel	**DH67BLB3**	5.14	Works Great with i7-2600k processor
Intel	**DP55WG**	4.6	multi-cpu smp works great, onboard NIC not supported by RouterOS 4.6 yet though, must use pci/pci express nics
Intel	**DQ965GFEKR (D41676)**	3.7, 3.4	Works fine on 3.7/3.4 if multi-cpu=no, BUT 3.5-3.7 fail to boot with E4600 processor and multi-cpu=yes
Intel	**S3210SHLC**	4.2	Boot from USB stick. Work fine for my PPPoE server. Up to 1300 users with summary 200Mbit traffic.
Intel	**D945GZT-M**	3.0rc4	Works fine
Chaintech	**AADF950**	3.0 beta 5	Works fine
Abit	**KT7E**	3.0 beta 7	Works fine
ECS	**nForce3-A**	3.6	Work fine including integrated ethernet
ECS	**P4M800PRO-M478**	2.9.43	No Apparent Problems, Disabled any unneeded devices in the BIOS
Abit	**BE6**	2.9.43	Works
VIA	**EPIA-MII12000**	2.9.42	Locks up under heavy load across wireless
Supermicro	**5015M-MR**	2.9.28	Motherboard is PDSMi w dual core
Supermicro	**PDSBM-LN2+**	2.9.51 & 3.16	[2]
Supermicro	**PDSMi-LN4+**	2.9.51, 3.13, 3.20	Very stable even with dual-cores enabled.
Gigabyte	**GA-41M-ES2L**	3.28	Works fine; CPU Intel Core2 Dual 2.7GHz; 2XRB44GV
Gigabyte	**GA-6BXS**	2.9.43	Works fine
Gigabyte	**GA-8I848P-G**	3.6	Work fine including integrated ethernet
Gigabyte	**GA-8ST667 rev. 3.0**	3.13	Works fine; CPU Intel Celeron 2,4GHz; Chipset SiS 645DX; 5xPCI;
Gigabyte	**GA-M720-US3 rev 1.0**	4.6	Works fine (downvolted to 1.1V)
Gigabyte	**GA-MA790GP-UD4H**	3.30	Works fine
Gigabyte	**K8-NS-ULTRA**	2.9.x-3.7	Excelent work, including both onboard ethernet (100 and 1000 lan)
Gigabyte	**GA-MA770-DS3 (rev. 2.0)**	3.10	Works fine and extreme stable include onboard LAN, IDE DOM can load normally
	Chipset P35 (Tested Using Gigabyte GA-P35-DS3L & Abit IP35)	3.7	Work fine but only with SATA, not IDE (Include DOM), bellow v3.7 problem with SATA too
Microsoft	**Virtual PC 2007**	3.7	Installs and and tries to boot.
VMware	**Workstation v6.0.3**	3.7	Runs Wicked Fast! I have had up to 8 Ethernet interfaces running simultaneously.
VMware	**ESXi v4**	3.30	Select IDE type for virtual disk - works perfectly!

	Xen 3.2.1 on Intel C2Q	4.x	Installs and runs fine on HVM bootloader using Intel VT technology. Even switches to RouterOS console from Dom0 shell. Ethernet interfaces work perfectly. **Do not install xen/kvm RouterOS packages!**
DFI	**AD73 Pro (Chipset VIA KT266A/VT8233ACD)**	2.9.x-3.30	Works fine. All 5 PCI's ocupied with 1 x LAN and 6 x R52H's (3 in RouterBoard11 and 3 in RouterBoard14!)
DFI	**AK75-EC (Chipset VIA KT133A/686B)**	3.14-3.30	Stable. All PCI's ocupied with LAN, miniPCI-PCI adapters fitted with R5H/R52H's and XR5's
Fujitsu Siemens	**Primergy RX100S5**	3.1, 3.7, 3.22	Can Install from Netinstall with RAID (LSI) Controller enable. Can't install from CD and Netinstall with only SATA or PATA mode. **But NOT RUN**
MSI	**785GM-E51**	4.11	Works fine, booted from USB stick, integrated LAN working

Ethernet chipsets

Vendor	Model	ROS version	Result
3Com	**3c905B Cyclone 100BaseTX**	2.9.51	Works! Extremely reliable, doesn't fully support tagged vlans
3Com	**3cSOHO100-TX [Hurricane] (rev: 48)**	3.14rc1	Works
3Com	**3c905C-TX/TX-M [Tornado] (rev: 120)**	2.9.51	Works! Extremely reliable, doesn't fully support tagged vlans
3Com	**3c905C-TX/TX-M [Tornado] (rev: 116)**	2.9.51	Works! Extremely reliable, doesn't fully support tagged vlans
3Com	**3c905B-FX Fast Etherlink XL FX 100baseFx [Cyclone] (rev: 0)**	2.9.43	Works but no link in Winbox and no Graph in Dude !!!!
Adaptec	**ANA-6944A/TX Quad 10/100MBit**	>=3.20	Works
Compaq	**NC3122 Fast Ethernet Server Adapter**	3.30	recognized but not working
Intel	**S82557 10/100MBit**	2.9.43	works
Intel	**PWLA8391GT PRO1000/GT**	3.7, 3.1, 2.9	Extremely Stable, used for years
Intel	**PWLA8391GTL PRO1000/GT**	3.4	Extremely Stable, used for years
Intel	**82575EB & 8257GB**	3.15	Added support
Intel	**82576 Gigabit ET Quad port**	4.5	Not recognized
Intel	**82576 Gigabit Dual Port (e1g42et)**	5.8	Works
Intel	**82572EI (EXPI9300PTBLK)**	4.5, 4.6	Works
Intel	**82572GI (EXPI9400PTBLK)**	4.5, 5.14	Works
Intel	**82574L (EXPI9301CT)**	5.14	Supported in ROS 5 / Works
Intel	**82571EB (EXPI9404PT) QUAD COPPER**	4.6, 5.14	Works
Intel	**S82557/S82555 10/100 Mbit TX**	2.9.50 / 3.16	Works Stable! FCC ID:EJMNPDSPD035
Intel	**PRO 1000 MT**	2.9.51	Works
Intel	**82541GI/PI Gigabit Ethernet Controller (rev: 5)**	2.9.49	Working but with high traffic (>100M) and many packets makes drops
Intel	**10Gbit Ethernet PCI Express**	3.17	Works
Intel	**82557/8/9 Ethernet Pro 100 (rev: 5)/Dual ports(Two ports/2-port)/RJ-45"**	4.10	Works, fine!

Intel	**82599ES - Intel X520 series adapters**	5.x & 6.x	Works, fine! Note: has SFP transceiver vendor restrictions: [3]
D-Link	**DFE-528TX rev. E1**	3.13	Works
D-Link	**DFE-580TX 4-port**	3.0 beta 5	Bad card, not recommended. Hangs router
D-Link	**DFE-530/538TX**	2.9.43 - 3.x	Works well, no apparent problems.
D-Link	**DUB-E100 USB**	3.18	added support, reported to be working
Marvell	**88E8001 Gigabit Ethernet Controller (rev: 20)**	3.13	Works
Marvell	**88E8056**	3.6	reported to be working with some BIOS setting enabled
DECchip	**21143 (ZYNX ZX410 4-port cPCI)**	2.9.51	Working
Realtek	**RTL-8169 Gigabit Ethernet x4 (rev: 16)**	3.0-3.11	Working Extremely reliable, used 4 mounts
Realtek	**RTL8111 (10/100/1000Mbit)**	3.10	Seems to be working only in older RouterOS v3 releases, v3.10 and before.
Realtek	**RTL8111C, RTL8111DL (10/100/1000Mbit)**	4.6 - 4.11	Working
Realtek	**RTL8139C+**	3.14	Works
Realtek	**RTL8139D**	3.x	Some work, others don't. Check for yourself.
Realtek	**RTL-8139/8139C/8139C+ (rev: 16)**	4.9 & 4.10	Works, fine!
Realtek	**RTL-8029(AS) (rev: 0)"**	4.9 & 4.10	Works, fine!
ZNYX	**ZX346Q**	3.27	Works
VIA	**VT6102 [Rhine-II] (rev: 67)"**	4.9 & 4.10	Works, fine!

x86 Systems

Model	ROS version	Result
Dell Optiplex GX1	2.9.x-3.0	Intel onboard/cpu 450-600Mhz, eth:3com, best for wirless stations; uptime over 200d, no problems at all.
Compaq Presario 2282	2.9.43	With 3c905 [Boomerang], no apparent problems
Dell GX100	2.9.x - 3.7	Intel onboard, 2 free pci, Intel cpu
Dell GX240	2.9.x - 3.7	Intel onboard, 2 free pci, Intel cpu, IDE HDD.
Dell GX260	2.9.x - 3.7	Intel onboard, 2 free pci, Intel cpu, IDE HDD.
Dell GX270	2.9.x - 3.7	Intel onboard, 2 free pci, Intel cpu. IDE HDD.
Dell GX280	2.9.x - 3.7	Intel onboard, 2 free pci, Intel cpu, SATA HDD.
Dell Dimension XPS GEN 3	2.9.x	Lan Onboard, 4 free pci, Intel cpu , SATA HDD (Excellent Stability)
Dell Inspiron Desktop 518	2.9.x - 3.7	After netinsall stuck on "loading system"
Dell PowerEdge 860	3.x	1U Rackmount, 2x Broadcom Gigabit onboard, 1x Intel CPU (many options), 1 PCI/1 PCIe or 2 PCIe riser options, SATA HDD ONLY. (some issues with floppy netinstall)
Dell PowerEdge R200	>= 3.19 recommended	Severe stability and clock issues with non-current ROS. Works like a top on 3.19 though. Also, if using an SATA-to-CF converter, the license key for the CF card in an R200 will only transfer to other R200's without Mikrotik reissuing it.

Dell PowerEdge R210	5.1 - 5.6	1U Rackmount. Half-depth (39cm) chassis. Dual port on-board Broadcom 5716 Gigabit Ethernet controllers. Single CPU on Intel 3420 Motherboard Chipset. Works OK and stable, once installed. Some issues with NetInstall - PXE boot works OK but install can't continue (says waiting for drivers...). Tested with Intel Gigabit ET Quad Port Server Adapter - works perfectly. - With Processador Intel E3-1220 and Broadcom NetXtreme II 5709 Gigabit NIC w/TOE & iSOE,Quad Port, Copper, PCIe-4 works fine.
Dell PowerEdge R310	5.0rc7	1U Rackmount. 2 x on-board Broadcom 5716 Gigabit Ethernet controllers. Single CPU on Intel 3420 Motherboard Chipset. Works OK and stable, once installed. Some issues with NetInstall - PXE boot works OK but install can't continue (says waiting for drivers...). Tested with Intel Gigabit ET Quad Port Server Adapter - works perfectly.
Dell PowerEdge 2950	3.x	2U Rackmount, Optional Redundant Power Supplies, 2x Broadcom Gigabit onboard, 2x Intel CPU (many options), 2- 8xPCIe & 1- 4xPCIe Standard (other risers available), SATA HDD ONLY, 2x internal USB - MUST SPECIAL ORDER WITHOUT RAID CONTROLLER. (some issues with floppy netinstall)}
HP Proliant DL380 G5	3.17	Works, but only if installed from CDROM (Netinstall to Windows mounted HDD causes issues)
HP Proliant ML110 G7	5.20	Works, installed from CD-ROM
Asus EEE PC 701	3.x	SFF Laptop, 1x10/100 ethernet (Not detected), Stock Wireless unsupported (AR5007E In Mini-PCIX slot), 630/900Mhz processor, 512MB RAM, 4GB SSD (Not detected), USB2.0 Bootable, SDHC Reader functions as a USB Stick
Dell PowerEdge SC1425	3.x	Rackmount, Intel Xeon 2.8 1MB 800FSB, 1024MB DDR2 PC3200 ECC, 2x Intel 82541GI Gigabit Ethernet, HD150gb SATA, USB works, very stable
Fujitsu Siemens Primergy RX100S5	3.7, 3.22	Can Install from Netinstall with RAID (LSI) Controller enable. Can't install from CD and Netinstall with only SATA or PATA mode. **But NOT RUN**
Toshiba Magnia SG20	2.9.44	CPU Celeron, VIA chipsets, onboard LAN Realtek and Intel, IDE HDD, PCMCIA tested with Orinoco Silver, miniPCI LT WinModem **not work**
Advantech FWA-3800	4.11, 4.17, 5.14	CPU Intel Core2 Duo 2,93GHz, 2GB RAM DDR2, 6 x Intel 1Gbps PCIe NIC, 1U size, works good as a BGP router. If you have troubles with MT 5.x on it, try to reset BIOS (by battery remove). Enable ACPI in the BIOS. It is disabled by default in Power Management Setup menu. Without this system does not boot (MT 5.x).
SuperMicro SYS-5015A-EHF-D525	5.18	CPU Intel Atom D525, 2x Intel 82574L 10/100/1000 Ethernet

Embedded Controllers

Model	ROS version	Result
WRAP.1E-2	2.9.51, 3.7	3 Ethernet, 1 miniPCI, 128 MB - Working
WRAP.2E	2.9.51, 3.7, 3.9, 3.10	1 Ethernet, 2 MiniPCI, 128 MB - Work fine
ALIX 2-2	2.9.48	2 Ethernet, 2 miniPCI - Working
Adlink cPCI-6770 Low Power Pentium III	2.9 and 3.0	CompactPCI CPU Module - Working, excellent performance!
Advantech PCA-6751	2.9.49	Working
Soekris 4801-50	2.9.48	3 Ethernet, 128MB, CF 512MB - Working
Soekris 4826-48	3.10	233 Mhz CPU, 128 Mbyte SDRAM, 1 Ethernet, 1 Serial, 256 Mbyte CF Flash, 2 Mini-PCI sockets, PoE. Limited power available/runs only 1 high power card (@26dB) along with another lower power card (@17dB)

Soekris net4801-48/50 + lan1641	3.22	All 7 (3+4) ethernet works, USB works (tested with Huawei 3G modem), extra serial port works. And RouterOS installed on CF card.
ALIX 2C0	2.9,3.0	2 Ethernet, 2 miniPCI ,128Mb 433Mhz Amd Geode- Working Perfect
ALIX 2C1	2.9,3.0	2 Ethernet, 2 miniPCI ,128Mb 433Mhz Amd Geode- Working Perfect
ALIX 2C3	2.9,3.0	2 Ethernet, 2 miniPCI ,256Mb 500Mhz Amd Geode- Working Perfect
ALIX 3C1	2.9,3.0	2 Ethernet, 2 miniPCI ,128Mb 433Mhz Amd Geode- Working Perfect
ALIX 3C2	2.9,3.0	2 Ethernet, 2 miniPCI ,256Mb 500Mhz Amd Geode- Working Perfect
ALIX 2D13	5.2	3 Ethernet, 1 miniPCI ,256Mb 500Mhz Amd Geode- Working Perfect

3G cards

Model	Tested RouterOS version	Comments	Format
AirPrime/Sierra PC 5220!	v3.x and higher		PCMCIA
Alcatel One Touch X020X USB (aka Longcheer WM66; Nuton NT36HD; MWalker mbd 100hu; Novacom GNS-3.5G White, SU-8200U; MTE MW610?)	v5.10 and higher	Config like Option_Globetrotter_HSDPA_USB_Modem Connected to Internet, Did not test Speed + Reliability (Alcatel OT X020X on x86) (data 0, info channel: 2)	USB
AnyData ADU E100A (aka "USB Wireless HSDPA/UMTS 2.1GHz GSM/GPRS/EGPRS 900/17000MHz/CDMA 1x EVDO Rev.A")	v3.x and higher		USB
AnyData ADU 500A USB (aka "USB Wireless HSDPA/UMTS 2.1GHz GSM/GPRS/EGPRS 900/1800MHz/CDMA 1x EVDO Rev.A")	v3.x and higher		USB
Audiovox PC5220 CDMA Dual Band 1XEV-DO PC Card	v3.x and higher		PCMCIA
C-motech CNU-680 CDMA 1x EV-DO 450Mhz USB Modem (used by Triatel) [4]	v3.x and higher		USB
Dell 5520	v3.x and higher		MiniPCI-E
Dell Wireless 5530 HSPA	v6.1 and higher	Data channel 0, Info channel 0, init: AT+CFUN=1 (needs manualy change profile by command AT*ENAP=1,1)	MiniPCI-E
Ericsson_F3507g_Mobile_Broadband_Module [5]	v3.x and higher	Set init string AT+CFUN=1, data channel and info channel to 3.	MiniPCI-e
Huawei E226 USB modem,	v3.x and higher		USB
Huawei E220 USB modem, E200BIS [6]	v3.x and higher		USB
Huawei E169 USB modem (used by Tele2) [7]	v3.x and higher		USB
Huawei E180 USB modem	v3.x and higher		USB
Huawei E1553 USB modem	v3.x and higher		USB

Huawei E1550 [8]	v3.x and higher		USB
Huawei E1762 USB Modem	v5.14 and higher	Locks up occasionally on 433UAH. Need to unplug to reset.	USB
Huawei E372 (USB) Videotron Canada	v5.15 and higher	Data channel 0 , Info channel 0, APN ihvm.videotron, Phone = *99# , Dial = ATDT ,	USB
Siemens M20	v3.x and higher		
Huawei E620;	v3.x and higher		PCMCIA
Kyocera KPC650	v3.x and higher		PCMCIA
Nokia CS-17 (USB)	v5.0 and higher	Data channel=2, info channel=4	USB
Nokia CS-18 (USB) Rogers Canada	5.12 and higher	Data channel=1, Info channel =1, APN= internet.com, Phone = *99# , Dial = ATDT , pap , Tested on rb-751 and rb-493	USB
Novatel EU740	v3.x and higher		MiniPCI-e
Novatel EU870 [9]	v3.x and higher		MiniPCI-e
Novatel MIFI 2372 Bell Canada	v5.12 and higher	Data Channel=0 , Info Channel= 0, APN = pda2.bell.ca , Phone = *99# , Dial = ATDT , pap, Tested on Rb-750UP and RB-493	USB
Novatel EV620 CDMA/EV-DO	v3.x and higher		MiniPCI-e
Novatel Merlin ES620 / Merlin ES720 / Ovation U720 [10]	v3.x and higher		USB
Novatel Merlin ES620	v3.x and higher		
Novatel Merlin S720 (HSDPA) [11]	v3.x and higher		PCMCIA
Novatel Merlin XU870 HSDPA/3G [12]	v3.x and higher		ExpressCard
Novatel U720 Wireless CDMA Modem	v4.5 and higher		USB
Novatel U730 (Wireless HSDPA Modem) [13]	v3.x and higher		PCMCIA
Novatel Wireless CDMA card	v3.x and higher		
Option Fusion UMTS Quad-GPRS	v3.x and higher		PCMCIA
Option Globetrotter HSDPA USB aka Teltonika ModemUSB/H7.2 (U3G150) [14]	v3.x and higher		USB
Probably most of the cards needing the HSO driver on Linux, tested: Option Globetrotter HSDPA USB (Globetrotter iCon 225 [15]	v5.12	Option_Globetrotter_HSDPA_USB_Modem see Workaround for Globetrotter devices offering no modem interface	USB

Option Qualcomm 3G WCDMA Model M00201-10886 (GTM378) [16]	v3.x and higher		miniPCI-e
Option Qualcomm 3G CDMA Model M00301 (GTM380) [17]	v3.x and higher	Set data channel and info channel to 3.	miniPCI-e
Option Qualcomm 3G CDMA Model M00401 (GTM382) [18]	v3.x and higher		miniPCI-e
Ericsson 3G F3607gw miniPCI-e	v3.x and higher	Set data channel and info channel to 2, set init string AT+CFUN=1	miniPCI-e
Sierra Aircard 595 [19]	v3.x and higher		PCMCIA
Sierra Aircard 595U USB Sprint Card [20]	v3.x and higher		USB
Sierra Wireless USB 306	v5.9 and higher	Data & Info Channel 2. For Telecom NZ use APN internet.telecom.co.nz and Phone number *99#	
Sierra Wireless USB 308 or AT&T Shockwave [21]	v5.0rc11 and higher	AT Commands are sent through Data Channel 2 or 3.	USB
Sierra Wireless AirCard 312U [22]	v5.2 and higher		USB
Sierra Wireless AirCard 320U [23]	v5.2 and higher		USB
Sierra Wireless AirCard 580 [24]	v3.x and higher		PCMCIA
Sierra Wireless AirCard 595 [19]	v3.x and higher		PCMCIA
Sierra Wireless AirCard 597E [25]	v3.x and higher		ExpressCard
Sierra Wireless AirCard 875 [26]	v3.x and higher		PCMCIA
Sierra Wireless AirCard 880 [27]	v3.x and higher		PCMCIA
Sierra Wireless AirCard 880 E [28]	v3.x and higher		ExpressCard
Sierra Wireless AirCard 881 [27]	v3.x and higher		PCMCIA
Sierra Wireless AirCard 881 E [28]	v3.x and higher		ExpressCard
Sierra Wireless EM5625 [29]	v3.x and higher		MiniPCI-e
Sierra Wireless MC5720 [30]	v3.x and higher		MiniPCI-e
Sierra Wireless MC5725 [31]	v3.x and higher		MiniPCI-e
Sierra Wireless MC8705 [32]	v5.1 and higher		MiniPCI-e
Sierra Wireless MC8755 [33]	v3.x and higher		

Sierra Wireless MC8755 for Europe [33]	v3.x and higher		MiniPCI-e
Sierra Wireless MC8765 [33]	v3.x and higher		MiniPCI-e
Sierra Wireless MC8775 [34]	v3.x and higher		MiniPCI-e
Sierra Wireless MC8780 [35]	v3.x and higher		MiniPCI-e
Sierra Wireless MC5725 [31]	v3.x and higher		MiniPCI-e
Sierra Wireless MC5727 [36]	v3.x and higher		MiniPCI-e
Sierra Wireless MC8785	v3.x and higher		MiniPCI-e
Sierra Wireless MC8790 [37]	v3.x and higher	Few models do not send echo for input commands, modem does not work properly.	MiniPCI-e
Sierra Wireless MC8792 [37]	v5.2 and higher	Info works only in channel 3. Channels 4 and 5 has limited AT set. datachannel=4, infochannel=3	MiniPCI-e
Sierra Wireless MC8781 [35]	v3.x and higher		MiniPCI-e
Sierra Wireless Sierra 598 (Sprint) USB [38]	v3.x and higher		USB
Sierra Wireless MP3G - EVDO	v3.x and higher		
Sierra Wireless MP3G - UMTS/HSPA	v3.x and higher		
Sierra Wireless Compass 885 (USB) [39]	v3.x and higher		USB
Silicon Labs MobiData GPRS USB Modem	v3.x and higher		USB
Sprint U301/301U 4G wireless card [40]	v4.6 and higher	C-MOTECH Co, FW301DOWMX, QUALCOMM Patch 33504--Tested with v4.11 on RB433UAH, Data CH=1 Info CH=3 Phone #777 for Sprint in US	USB
Sprint U300/300U 4G wireless card [41]	v4.6 and higher	C-MOTECH Co, FW301DOWMX, QUALCOMM Patch 33504	USB
Franklin M600 3G/4G wireless card [42]	v5.x and higher	only 3G mode	MiniPCI-e
Verizon Express Network PC5220 (AirPrime 5220)	v3.x and higher		
ZTE AC8700	v3.x and higher		
ZTE MF620 / MF622 [43]	v3.x and higher		USB
ZTE MF620 / MF622 (3G) [43]	v3.x and higher		USB
ZTE MF100 [44]	v5.11 and higher	Set info channel = 1, data channel = 2	USB

ZTE MF680 [45]	v5.4 and higher	Used by 3 in Sweden. Set data chanel to 1	USB
ZTE MF668 [46]	v4.5 and higher	for Rogers Wireless (Canada) Set APN: isp.apn and Info & Data Channel to 1	USB
T-Mobile (Germany) Web´n´Walk Box Micro (Huawei E220) [6]	v3.x and higher		USB
Vodafone (Germany) Easybox 2 (Huawei E220) [6]	v3.x and higher		USB
Surfbox Mini (Huawei E220) [6]	v3.x and higher		USB
E-Plus & Base (Germany) USB Minimodem (Huawei E220) [6]	v3.x and higher		USB
Huawei E600	v3.x and higher		PCMCIA
Novatel Merlin V640/XV620 [47]	v3.x and higher		ExpressCard
Novatel Merlin V620/S620 [48]	v3.x and higher		PCMCIA
Novatel Merlin EX720/V740/X720 [49]	v3.x and higher		ExpressCard
Novatel Merlin V720/S720/PC720 [50]	v3.x and higher		PCMCIA
Novatel Merlin XU870 HSDPA/3G [51]	v3.x and higher		ExpressCard
Novatel X950D [52]	v3.x and higher		ExpressCard
Novatel ES620/ES720/U720/USB720	v3.x and higher		USB
Novatel E725/E726 [53]	v3.x and higher		MiniPCI-e
Vodafone EU740/Novatel non-Vodafone EU740	v3.x and higher		MiniPCI-e
Vodafone K3565/Huawei E160 [54]	v3.x and higher		USB
Novatel EU850D/EU860D/EU870D [55]	v3.x and higher		MiniPCI-e
Novatel MC930D/MC950D [56]	v3.x and higher		USB
Novatel MC727/U727 [57]	v3.x and higher		USB
Novatel Expedite EV620 CDMA/EV-DO	v3.x and higher		MiniPCI-e
Novatel Expedite EU740 HSDPA/3G, Dell Wireless 5500 Mobile/Dell Wireless 5505 Mobile	v3.x and higher		MiniPCI-e
Novatel Expedite E720 CDMA/EV-DO	v3.x and higher		MiniPCI-e

Novatel Expedite ET620 CDMA/EV-DO	v3.x and higher		PCMCIA
Onda H600/ZTE MF330	v3.x and higher		
Onda MDC525UP		Not supported	USB
Onda MT833UP (opt. ext. antenna)	v5.6 and higher	Set info channel = 1, data channel = 0	USB
Onda MT835UP (opt. ext. antenna)	v5.21 and higher	Set info channel = 1, data channel = 0	USB
BP3-USB & BP3-EXT HSDPA	v3.x and higher		USB
ZTE MY 39 (MSM 6500 based) [58]	v3.x and higher		PCMCIA
Cricket A600	v3.x and higher		
Globetrotter HSDPA Modem Option N.V.	v3.x and higher		
Sony Ericsson MD300	v3.x and higher		
ZTE MF 626 [59]	v3.x and higher		USB
ZTE MF 627 [60]	v3.x and higher		USB
Pantech / UTStarcom UM175	v3.x and higher		USB
Novatel U760 [61]	v3.x and higher		USB
ZTE K3565-Z [62]	v4.4 and higher	Revision: BD_P673A2V1.0.0B09	USB
Novatel Expedite EV620	v4.5 and higher		MiniPCI-e
Novatel MC760 VMU [63]	v4.5 adn higher		USB
Franklin Wireless FW300DOWMX	v4.5 and higher		
Huawei EC1260	v4.5 and higher		USB
Vodafone K3520-Z [64]	v4.6 and higher		USB
Vodafone K3765 [65]	v4.6 and higher		USB
Telstra 3G Elite	v5.x and higher		USB
Vodafone Huawei K4505 [66]	v5.x and higher	Data-channel=0 Info-channel=3	USB
Vertex VW 110	v5.x and higher		

ZTE MF112 [67]	v5.x and higher	Power issues on mipsbe boards	USB
Huawei ET127	v5.x and higher		3G
Huawei EC1261	v5.x and higher		USB
Huawei E173 [68]	v5.x and higher		USB
ZTE MF190 [69]	v5.x and higher	Data channel=3 and info Channel=1	USB
ZTE MF102 [70]	v5.x and higher	Works! Possible that need to change data channel=2 and info channel=2	USB
Option Globetrotter GT380	v5.x and higher		
Simcom 5220	v5.x and higher		
Huawei K3770	v5.x and higher		
Novatel USB551L (Verizon) [71]	v5.9 and higher	Only 3G support (No LTE support)	UB
Novatel Wireless MIFI4510		Not supported	
ZTE MC2718 [72]	v5.8 and higher	Possible data-channel=0 info-channel=1 GPS(NMEA)=4	MiniPCI-e
LG-VL600 (Verizon)		Not supported	
Huawei EC156 [73]	v5.8 and higher		USB
K3806 [74]	v5.8 and higher		USB
ZTE MF-210V [75]	v5.9 and higher		MiniPCI-e
Huawei E398 [76]	v5.9 and higher		USB
Huawei E367	v5.11 and higher		USB
Huawei EM770 [77]	v5.11 and higher		MiniPCI-e
Nokia E52 (Series 60)	v5.12 and higher	Set Usb mode to "PC Suite" in phone menu, Baud rate - 115200, Ports - 0/0, APN - internet, no modem init string, Dial number - *99#	USB
Nokia 6700 Classic (Series 40)	v5.12 and higher	Set Usb mode to "PC Suite" in phone menu Baud rate - 115200 Ports - 0/0 APN - internet Modem init string - ATZ, Dial number - *99# Phone is charged up through the USB port	USB
Alcatel X220S [78]	v5.13 and higher		USB
MO835UP	v5.14 and higher		USB

Nokia Datacard CS-11 & CS-15	v5.14 and higher		USB
ZTE MF821	v5.15 and higher		USB
Huawei K4510 [79]	v5.15 and higher		USB
Huawei E173s	v5.15 and higher		USB
Huawei E352		Not supported	USB
Option3G Mini-PCI model GTM661W		Not supported	USB
Option 225	v5.15 and higher		USB
ZTE AC682		might not work	USB
ZTE AX320	v5.15 and higher		USB
ZTE MF 652	v5.15 and higher		USB
Vodafone Huawei K-4605	v5.15 and higher		USB
Huawei E353	v5.15 and higher	Some revisions might not work.	USB
CELOT CT-680	v5.16 and higher		USB
Huawei / ZTE 669T [80]	v5.8 and higher	Info/data =1 and 2 worked	USB
Huawei / ZTE MF195 [81]	v5.18 (maybe 5.17, but 5.16 not worked)	Info/data =1 worked	USB
Haier CE81B	v5.21		USB
Huawei E1731	v5.21 and higher		USB
Huawei EM820W [82]	v5.11 and higher	Data channel=0, Info channel=2, Keep info channel open	MiniPCI-e
Alcatel X221L [83]	v5.20 and higher		USB
Telecom NZ T-Stick ZTE MF-181	v6.0rc13	Data Channel=2, Info Channel=2, APN internet.telecom.co.nz, PHONE=*99#. Tested ok for both data and SMS on CCR1016-12G	USB

***** - Currently MikroTik RouterOS works with PPP 3G modems over serial interfaces, 4G modems with IP drivers are not supported.

4G LTE cards

LTE should be configured under the new "/interface lte" menu

Model	Tested RouterOS version	Comments	Format
BandRich C501 [84]	v5.25 and v6.0		USB
Sierra Wireless MC7710 [85]	v5.25 and v6.0	If modem uses firmware 3.5 it should be upgraded to 3.5.23.2 firmware release in order to work in RouterOS correctly again.	MiniPCI-e
Sierra Wireless AirCard 320U [86]	v6.0	Vendor/product id pair should be 0x0f3d/0x68AA and DirectIP firmware loaded on the modem.	USB
Yota LU150 [87]	v5.22 and v6.4	Some settings are ignored. Works in Russian markets.	USB
Yota WLTUBA-107 [88]	v6.0	Some settings are ignored. Works in Russian markets.	USB
Yota wifi modem [89]	v6.7	Some settings are ignored. Works in Russian markets.	USB
Vodafone K4305 [90]	v6.7	Some settings are ignored.	USB
Android usb tethering interface	v6.7	Some settings are ignored.	USB
ZTE MF823 [91]	v6.8	Some settings are ignored.	USB

Memory cards

NB! New flash cards need always formatting!

Legend:

- V - works
- X - doesn't work
- ? - not tested
- NA - not available for this RB

Model	Type	RB600	RB1000	RB433AH	RB450G	R493G
PQI 4GB 120x HiSpeed	CF	**X**	?	NA	?	?
2GB Transcend 133x	CF	X	?	NA	?	?
2GB Kingston 133x (Elite Pro, code CF/2GB-S2)	CF	V	?	NA	?	?
4GB Kingston 133x (Elite Pro)	CF	?	V	NA	?	?
4GB Kingston CF/4GBIN	CF	**X**	?	NA	?	?
8GB A-Data Speedy G08GNMC7B0095	CF	V	V	NA	?	?
16GB A-Data Speedy	CF	V	V	NA	?	?
4GB Apacer Photo Steno IV 300X	CF	**X**	**X**	NA	?	?
1GB Sandisk Ultra II BB05024GA	CF	V	V	NA	?	?
2GB Sandisk Ultra II	CF	?	V	NA	?	?
8GB Sandisk Ultra II	CF	?	V	NA	?	?
8GB SanDisk Extreme III (200X,30MB/s)	CF	?	V	NA	?	?
2.5GB Seagate ST1 (ST625211CF)	CF microdrive	V	V	NA	?	?

8GB Seagate ST1	CF microdrive	V	V	NA	?	?
64MB Nokia	microSD	NA	NA	?	V	?
512MB Kingston	microSD	NA	NA	V	?	?
1GB Apacer	microSD	NA	NA	V	?	?
1GB Kingston (SDC/1GB)	microSD	NA	NA	?	X	?
2GB Kingston	microSD	NA	NA	**X**	**X**	?
2GB Traxdata	microSD	NA	NA	?	**V**	?
4GB Apacer SDHC (class 6)	microSD	NA	NA	V	X	?
4GB Axiz	microSD	NA	NA	V	?	?
4GB Kingston	microSD	NA	NA	V	?	?
4GB Kingston SDHC (C04G JAPAN class 4)	microSD	NA	NA	V	?	?
4GB Maxell SDHC (class 2) P-series	microSD	NA	NA	?	?	V
4GB Transcend SDHC	microSD	NA	NA	V	?	?
4GB Sandisk Mobile Ultra Micro SDHC (with card reader)	microSD	NA	NA	?	V	?
8GB Sandisk SDHC (0733702482DLE)	microSD	NA	NA	V	V	?
8GB Sandisk Mobile microSDHC (SDSDQ-8192-A11M)	microSD	NA	NA	?	V	?
8GB Sandisk Mobile Ultra SDHC (Class 6)	microSD	NA	NA	?	V	?
8GB Kingston SDHC (Class 4)	microSD	NA	NA	V	X	?
8GB ADATA micro SDHC (AUSDH8GCL6) (Class 6)	microSD	NA	NA	?	V	?
16GB Sandisk micro SDHC (SDSDQ-16384-P36M) class 2	microSD	NA	NA	NA	X	?
16GB Sandisk micro SDHC (0835B03279DQ) class 2	microSD	NA	NA	V	V	?
2GB Sandisk micro SDHC class 2	microSD	NA	NA	NA	X	V
4GB Sandisk Mobile microSDHC (SDSDQM-004G-B35S) class 4	microSD	NA	NA	?	X	X
2GB Transcend microSD (TS2GUSD) class not stated	microSD	NA	NA	?	X	?

Note: Pushing the "Kingston SDHC 8GB card" all the way into the socket caused the card not to work properly! It had to be pulled out ~1mm for it to work!

802.11a/b/g wireless cards

Model	Form factor	Platform	ROS	Result
RouterBOARD R52 [92]	miniPCI	ALL	ALL	Natively supported!
RouterBOARD R52H [93]	miniPCI	ALL	ALL	Natively supported!
TechnicLan TMP-5414A [94]	miniPCI	All RouterBOARDs	2.9 / 3.x / 4.x	Perfect, Stable
Compex WLM54AG-20 [95]	miniPCI	RB1xx / RB3xx / RB4xx / RB5xx / RB6xx / x86	2.9 / 3.x / 4.x	Perfect, Stable
Compex WLM54A-26 [96]	miniPCI	RB1xx / RB3xx / RB4xx / RB5xx / RB6xx / x86	2.9 / 3.x / 4.x	Perfect, Stable

SparkLAN WMIA-165G	miniPCI	RB1xx / RB3xx / RB4xx / RB5xx / RB6xx / x86	2.9 / 3.x / 4.x	Perfect, Stable
SparkLAN WMIA-166AG	miniPCI	RB1xx / RB3xx / RB4xx / RB5xx / RB6xx / x86	2.9 / 3.x / 4.x	Perfect, Stable
SparkLAN WMIA-166AGH	miniPCI	RB1xx / RB3xx / RB4xx / RB5xx / RB6xx / x86	2.9 / 3.x / 4.x	Perfect, Stable
Alfa AWPCI 085 H	miniPCI	RB1xx/RB333/RB4xx/x86	2.9&3.x	All just perfect
TP-Link TL-WN550/551	PCI	x86	2.9&3.x	Perfect 19dB rated, stable at 21
TP-Link TL-WN650/651	PCI	x86	2.9&3.x	Perfect 19dB rated, stable at 21. Unstable with compression activated
D-Link AG530 a/b/g (both rev.A1 and A2)	PCI	x86	2.9&3.x	Perfect. Works perfect, in both 2.4 and 5.x GHz
D-Link DWL-G510	PCI	x86	only 2.9.x tested	Perfect. Tested Rev A1
D-Link DWL-G520	PCI	x86	2.9.x & 3.x	Works well.
D-Link DWL-G520+A/RaLink 2591 Chipset	PCI	x86	4.9	NOT Work!
Gigabyte b/g GN-WI01GT	miniPCI-e	x86	3.x	Works also in RouterBOARDs with miniPCI-e slot
Senao NL-2511CD EXT2	PCMCIA	x86&rb230	only 2.9x tested	Perfect. Just about the most sensitive card i used, in the good sense. Only 11b.
Planet WL-8310	PCI	x86	only 2.9.x tested	Perfect. Stability.
Netgear WG311T 108	PCI	x86	only 2.9x tested	Perfect. stability
SMC SMCWPCIT-G	PCI	x86	3.x tested	Perfect on A/B/G.
Wistron DCMA-81	miniPCI	x86,rb	2.9.xx, 3.xx	Perfect on A/B/G with or without compression.
Wistron DCMA-82	miniPCI	rb	3.xx	Works on some RB, but on 2/3 of my RB433 it causes the board to reboot when enabled. Many other people have had similar experiences. Not recommended. Maybe OK on 133
Senao NMP-8602	miniPCI	x86&rb411	2.9x,3.x tested	Perfect on a/b/g
Dbii F20 [97]	miniPCI	RB411 & RB433	3.x tested	Perfect on B/G, too thick for 3 in rb433
Dbii F50 [98]	miniPCI	RB411 & RB433	3.x tested	Perfect on A, too thick for 3 in rb433
Dbii F20-PRO [99]	miniPCI	RB411 & RB433	3.x tested	Perfect on B/G, too thick for 3 in rb433
Dbii F50-PRO [100]	miniPCI	RB411 & RB433	3.x tested	Perfect on A, too thick for 3 in rb433
TP-Link TL-WN751ND [101]]	PCI	x86	v5.xx, v6.xx	Not Working!
TP-Link TL-WN851ND [102]]	PCI	x86	v5.xx, v6.xx	Working

TP-Link TL-WN881ND [102]]	PCI-e	x86	v5.xx, v6.xx	v5.xx NOT working not appear in interfaces, v6.x working (6.1 unstable after scan stop working in client or manual scan, set to AP-bridge mode working ok)
XR2	miniPCI	RB433	3.x	works
XR5	miniPCI	RB433, x86	3.x	works
SR9	miniPCI	RB433	3.x	works
SR9	miniPCI	RB411 & RB433	3.x	works
Valemount KXS30SG [103]	miniPCI	RB433	3.2 & 4.2 tested	B/G only, no A. Big heat sink means card should be installed on highest mount when other cards being used. High power consumption can cause power problems, otherwise works perfectly.
Zcomax XG-650 [104]	miniPCI	RB112 & RB433	4.2 & 4.6	Not working (driver doesnt work / exist). It has a TI TNETW1130GVF chipset.
AR5BMB5	miniPCI	RB411,x86	3.xx, 4.3	Not working. Atheros chipset.

802.11n wireless cards

Model	Form factor	ROS	Result
RouterBOARD R2N [105]	miniPCI	4.0b3	works
RouterBOARD R52N [105]	miniPCI	4.0b3	works
RouterBOARD R5H [106]	miniPCI	v4	works
SparkLAN WMIA-198N [107]	miniPCI	4x, 5x	Works
Compex WLM200N5-23 [108]	miniPCI	4.0b3	Works
Compex WLM200NX [109]	miniPCI	4.0b3	Works
Dbii F52N-PRO [110]	miniPCI	5.1	Works
TP-Link Wireless N (2T2R)	miniPCI	4.0b3	Works
Ubiquiti	miniPCI	4.0b3	Works
Ubiquiti	miniPCI	4.0b3	Works

USB wireless cards

Model	Form factor	ROS	Result
802 b/g/n AR9271	USB	5.0rc4	works
802 b/g/n AR9271 Ubiquiti WiFiStation	USB	5.8 x86	works on N but causes reboot on G
802 b/g/n AR9271 ALFA AWUS036NHA	USB	5.14 x86	works
802 b/g/n AR9170 NETGEAR WN111v2	USB	5.16 x86	Don't recognize

T1/E1

Model	Form factor	Platform	ROS	Result
Farsite FarSync TE1	PCI	x86	3.15	supported

Note: Since v3.15 RouterOS doesn't support any Sync/T1/E1 cards except select Farsite models

GPS

Model	Connection	Platform	ROS	Result
EXAMPLE	USB	x86	3.30	supported

Storage controllers (SAS/SCSI)

Post only tests since RouterOS v5beta5

Brand	Model	Motherboard	RouterOS	Works	Type
HP	Smart Array E200i	x86	v5rc1	No	SCSI
3ware	3w-9xxx	x86	v5rc8	Yes	SCSI
Areca	arcmsr	x86	v5rc8	Yes	SCSI
Megaraid	Megaraid (some Dell servers)	x86	v5rc8	Yes	SCSI

LCD panels

Brand	Model	Motherboard	RouterOS	Works / Doesn't
Crystalfontz	CFA-633v1.5	x86	v5RC10	Works
Crystalfontz	CFA-633v1.3	x86	v5RC7-RC10	Didn't work

USB storage

Note: USB storage device that does not require special drivers or is compatible to work with generic USB storage drivers will work. Devices include external hard drives, USB flash drives

Brand	Model	Works / Doesn't
generic	USB flash drive	Works
Kingston	DataTraveler 2GB	Works

SFP modules

Brand	Model	Rate	Connector/Cable type	Wavelength	Tested with	Works / Doesn't
MikroTik	S-85DLC05D	1,25G	Dual LC, MM	850	CCR/RB2011	Natively supported
MikroTik	S-31DLC20D	1,25G	Dual LC, SM	1310	CCR/RB2011	Natively supported
MikroTik	S-35LC20D	1,25G	LC, SM	T1310nm/R1550nm	CCR/RB2011	Natively supported
MikroTik	S-53LC20D	1,25G	LC, SM	T1550nm/R1310nm	CCR/RB2011	Natively supported
Finisar	FCLF-8521-3	10/100/1000	RJ45, Cat6	-	RB2011LS-IN	Works!
Finisar	FCLF-8521-3-MD	10/100/1000	RJ45, Cat6	-	RB2011LS-IN	Works!
Finisar	FTRJ8519P1BNL-B1	10/100/1000 1.25 Gb/s 1000Base-SX Ethernet & Dual Rate 1.063/2.125 Gb/s Fibre Channel	LC, MM	850	RB2011LS-IN	Works!
Finisar	FTLF8519P2BNL	10/100/1000 1.25 Gb/s 1000Base-SX Ethernet & Dual Rate 1.063/2.125 Gb/s Fibre Channel	LC, MM	850	RB2011LS-IN	Works!
Unica	SFP-1.25G-T	1000M	RJ45, Cat6	-	RB2011LS-IN	Works!
Dell	FTLX8571D3BCL	1,25G	LC, MM	850	RB2011LS-IN	Works!
Unica	GP-3124-L2CD-C	1,25G	LC, MM	1310	RB2011LS-IN	Works!
Cisco	SFP-10G-LR	10G	LC/PC, SM	1310	RB2011LS-IN	Works!
Cisco	GLC-T	1.25G	RJ45, Cat6	-	RB2011LS-IN	Works!
Cisco	GLC-SX-MM	1000BASE-SX SFP transceiver module for MMF, 1.25G	LC/PC, MM	850	RB2011LS-IN	Works!
6COM	6C-SFP-T	10/100/1000	RJ45, Cat6	-	RB2011LS-IN	Works!
6COM	6C-WDM-0210BSD	1,25G	Bi-Di SC, SM	1550/1310	RB2011LS-IN	Works!
6COM	6C-WDM-0210ASD	1,25G	Bi-Di SC, SM	1310/1550	RB2011LS-IN	Works!
6COM	6C-SFP-0310D	1,25G	LC, MM	1310	RB2011LS-IN	Works!
6COM	6C-SFP-0301D	1,25G	LC, MM	850	RB2011LS-IN	Works!
Ingellen	INSP-T(10/100/1000)	10/100/1000	RJ45, Cat6	-	RB2011LS-IN	Works!
Ingellen	INSPL-53-BX	1,25G	Bi-Di LC, MM	1550/1310	RB2011LS-IN	Works!
Ingellen	INSPL-35-BX	1,25G	Bi-Di LC, MM	1310/1550	RB2011LS-IN	Works!
Ingellen	INSP -LX-SM	1,25G	LC, MM/SM	1310	RB2011LS-IN	Works!

Ingellen	INSP-SX-MM	1,25G	LC, MM	850	RB2011LS-IN	Works!
AXCEN	AXGT-R1T4-05I1	10/100/1000	RJ45, Cat6	-	RB2011LS-IN	Works!
AXCEN	AXGD-37A4-0531	1,25G	Bi-Di LC, MM	1550/1310	RB2011LS-IN	Works!
AXCEN	AXGD-16A4-0531	1,25G	Bi-Di LC, MM	1310/1550	RB2011LS-IN	Works!
AXCEN	AXGD-1354-0531	1,25G	LC, MM	1310	RB2011LS-IN	Works!
AXCEN	AXGD-5854-0511	1,25G	LC, MM	850	RB2011LS-IN	Works!
TP-Link	TL-SM311LS	1,25G	LC, SM	1310	RB2011LS-IN	Works!
TP-Link	TL-SM311LM	1,25G	LC, MM	850	CCR1036 12G-4S	Works!
OPTIC	OPTIC-SFP-3524S-02-SC	1,25G	BiDi SC, SM	TX1310/RX1550	RB2011UAS-RM, RB260GS	Works!
OPTIC	OPTIC-SFP-5324S-02-SC	1,25G	BiDi SC, SM	TX1550/RX1310	RB2011UAS-RM, RB260GS	Works!
OPTIC	OPTIC-SFP-S1203-L3302-LC	1,25G	BiDi LC, SM	TX1310/RX1550	RB2011UAS-RM, RB260GS	Works!
OPTIC	OPTIC-SFP-S1205-L3302-LC	1,25G	BiDi LC, SM	TX1550/RX1310	RB2011UAS-RM, RB260GS	Works!

USB serial adapters

Note: when USB serial port is connected RouterOS might attach serial console on the port. Before using it for something else, disable the console on the interface

Brand	Model	Works / Doesn't
	USB U209-000-R Serial-Port	Works
	FT232RL chipset	Works

USB Ethernet

Note: see if device works with one of these linux kernel modules. If yes, it will be possible to use it on RouterOS

RouterOS on x86 have these modules enabled

- USB_PEGASUS
- USB_RTL8150
- USB_USBNET
- USB_NET_AX8817X
- USB_NET_CDCETHER
- USB_HSO

RouterOS on mips have these modules enabled:

- USB_NET_MCS7830
- USB_NET_AX8817X
- USB_NET_CDCETHER

- USB_HSO
- USB_USBNET

AX88178 (USB2.0 Gigabit Ethernet) recognized but not working.

References

[1] http://wiki.mikrotik.com/wiki/Manual:Driver_list
[2] http://forum.mikrotik.com/viewtopic.php?f=1&t=28184
[3] http://www.intel.com/support/network/adapter/pro100/sb/CS-030612.htm
[4] http://www.cmotech.com/eng/usbModems/product.do?act=view&product_seq=55
[5] http://3g-modem.wetpaint.com/page/Ericsson+F3507G
[6] http://www.huaweidevice.com/worldwide/productFeatures.do?pinfoId=282&directoryId=5008&treeId=582&tab=0
[7] http://3g-modem.wetpaint.com/page/Huawei+E169+%28E169G%2C+E169V%2C+K3520%29
[8] http://3g-modem.wetpaint.com/page/Huawei+E1550
[9] http://www.novatelwireless.com/content/pdf/EU870D_datasheet.pdf
[10] http://www.novatelwireless.com/index.php?option=com_content&view=article&id=177&Itemid=58
[11] http://support.sprint.com/support/device/Novatel_Wireless/Merlintrade_S720_by_Novatel_Wireless-novatel_s720
[12] http://www.novatelwireless.com/content/pdf/Merlin_XU870_DataSheet.pdf
[13] http://www.novatelwireless.com/content/pdf/Merlin_U730_Datasheet.pdf
[14] http://teltonika.lt/uploads/docs/ModemUSB%20H7.2%20User%20Manual%20EN.pdf
[15] http://www.option.com/en/products/products/usb-modems/icon225/
[16] http://www.wireless-market.hu/dwl/OPTION_GTM378_DS_ENG.pdf
[17] http://www.option.com/en/products/products/modules/gtm380e/
[18] http://www.option.com/en/products/products/modules/gtm382e/
[19] http://support.sprint.com/support/device/Sierra_Wireless/AirCard_595_by_Sierra_Wireless-sierra_ac595
[20] http://support.sprint.com/support/device/Sierra_Wireless/AirCard_595U_by_Sierra_Wireless-sierra_ac595u
[21] http://www.sierrawireless.com/productsandservices/~/media/Data%20Sheet/datasheet_aircard308-310u.ashx
[22] https://www.sierrawireless.com/productsandservices/AirCard/USBModems/aircard_312u.aspx
[23] http://www.netgear.com/service-provider/products/mobile-broadband/usb-modems/aircard_320U.aspx
[24] http://support.sprint.com/support/device/Sierra_Wireless/AirCard_580_by_Sierra_Wireless-sierra_ac580
[25] http://support.sprint.com/support/device/Sierra_Wireless/AirCard_597E_by_Sierra_Wireless-sierra_597e
[26] http://www.nucleusnetworks.co.uk/3g-data-card/docs/sierra_aircard_875_spec.pdf
[27] http://www.nucleusnetworks.co.uk/3g-data-card/docs/sierra_aircard_880_spec.pdf
[28] http://www.nucleusnetworks.co.uk/3g-data-card/docs/sierra_aircard_880E_1.1spec.pdf
[29] http://services.koretelematics.com/devices/images%5CDevices%5CSierra%20Wireless%5CEM-5625%5CEM-5625%20-%20SpecSheet.pdf
[30] http://www.hy-line.de/fileadmin/hy-line/communication/hersteller/Sierra_Wireless/Dokumente/Flyer_MC5720.pdf
[31] http://www.hy-line.de/fileadmin/hy-line/communication/PR/Text/Flyer_MC5725.pdf
[32] http://www.m2mconnectivity.com.au/sites/default/files/brochures/Sierra_Wireless_AirPrime_MC_Series_Intelligent_Embedded_Modules.pdf
[33] http://www.hy-line.de/fileadmin/hy-line/communication/hersteller/Sierra_Wireless/Dokumente/Flyer_MC8755_65.pdf
[34] http://3g-modem.wetpaint.com/page/Sierra+Wireless+MC8775+%26+MC8775v
[35] http://www.hy-line.de/fileadmin/hy-line/communication/hersteller/Sierra_Wireless/Dokumente/MC_8780_8781_Datasheet_hires_web.pdf
[36] http://www.rell.com/resources/RellDocuments/SYS_26/Sierra%20Wireless_MC5727.pdf
[37] http://www.sierrawireless.com/productsandservices/AirPrime/Wireless_Modules/High-speed/~/media/Data%20Sheet/AirPrime_datasheets/Sierra_Wireless_AirPrime_MC_Series_Intelligent_Embedded_Modules.ashx
[38] http://www.sierrawireless.com/product/~/media/Data%20Sheet/datasheet_aircardusb598.ashx
[39] http://www.insitefleet.com/documents/Compass_885_Datasheet_web.pdf
[40] http://support.sprint.com/support/device/Sprint/U301USB_Device_Sprint_3G4G_Mobile_Broadband-dvc1020001prd
[41] http://support.sprint.com/support/device/Sprint/3G4G_USB_Modem_U300-franklin_u300
[42] http://www.franklinwireless.com/image/ebrochure/M600_datasheet_v1.pdf
[43] http://wwwen.zte.com.cn/endata/mobile/UK/UK_Instruction/201011/P020101118724987299606.pdf
[44] http://www.3gmodem.com.hk/ZTE/MF100.html
[45] http://www.3gmodem.com.hk/ZTE/MF680.html
[46] http://wwwen.zte.com.cn/en/products/mobile/mobile_detail_291.jsp?mobileName=MF668
[47] http://www.novatelwireless.com/images/pdf/Merlin_XV620_Datasheet.pdf
[48] http://www.novatelwireless.com/content/pdf/Merlin_V620_Datasheet.pdf

[49] http://www.novatelwireless.com/images/pdf/Merlin_X720_Datasheet.pdf
[50] http://www.novatelwireless.com/content/pdf/Merlin_PC720_Datasheet.pdf
[51] http://www.novatelwireless.com/images/pdf/Merlin_XU870_DataSheet.pdf
[52] http://www.novatelwireless.com/content/pdf/Merlin_X950D_datasheet.pdf
[53] http://www.novatelwireless.com/content/pdf/Expedite_E725_Datasheet.pdf
[54] http://3g-modem.wetpaint.com/page/Huawei+E160+%28E160G%2C+E160E%2C+E160X%2C+K3565%29
[55] http://www.novatelwireless.com/content/pdf/EU850D_Datasheet.pdf
[56] http://www.novatelwireless.com/content/pdf/OvationMC950D_datasheet.pdf
[57] http://www.novatelwireless.com/content/pdf/Ovation_MC727_Datasheet.pdf
[58] http://ebookbrowse.com/manual-zte-my39-eng-pdf-d40571716
[59] http://3g-modem.wetpaint.com/page/ZTE+MF626
[60] http://3g-modem.wetpaint.com/page/ZTE+MF627
[61] http://www.novatelwireless.com/index.php?option=com_content&view=article&id=166
[62] http://3g-modem.wetpaint.com/page/ZTE+K3565-Z+%28Vodafone%29
[63] http://www.novatelwireless.com/content/pdf/Datasheet_MC760.pdf
[64] http://www.business.vodafone.com/download/getFmlDoc.do?docId=7c3f8af5-9fcf-41c4-8847-d4059a0665f0
[65] http://www.3gmodem.com.hk/Huawei/K3765.html
[66] http://www.twayf.com/huawei-k4505
[67] http://www.3gmodem.com.hk/ZTE/MF112.html
[68] http://3g-modem.wetpaint.com/page/Huawei+E173
[69] http://wwwen.zte.com.cn/endata/mobile/info/201102/t20110209_219190.html
[70] http://wwwen.zte.com.cn/en/products/mobile/mobile_detail_291.jsp?mobileName=ZTE%20MF102
[71] http://www.novatelwireless.com/content/pdf/MC551NVTLDatasheetRev2.pdf
[72] http://www.headele.com/Datasheet/EVDO/MC2716&MC2718%20Technical%20Specifications%20and%20Hardware%20Design.pdf
[73] http://www.tataphoton.com/download/user-manuals/Huawei-EC156-User-Manual.pdf
[74] http://vodafone.com/content/dam/vodafone/about/what/devices/mobile_broadband/pdf/vodafonek3806specs.pdf
[75] http://www.smartm2msolutions.com/Recursos/downloads/MF210.pdf
[76] http://www.twayf.com/huawei-e398-lte-modem
[77] http://www.arm9.net/datasheet/EM770.pdf
[78] http://www.alcatelonetouch.com/global-en/products/mobile_broadband/ot-x220.html
[79] http://www.vodafone.com/content/dam/vodafone/about/what/devices/mobile_broadband/pdf/vodafonek4510k4511specs.pdf
[80] http://www.3gmodem.com.hk/ZTE/MF669.html
[81] http://wwwen.zte.com.cn/endata/mobile/Hungary/Hungary_Instruction/201111/P020111102401309130495.pdf
[82] http://techship.se/products/huawei-em820w/
[83] http://www.germanos.bg/en/catalogue/mobile-communications/alcatel-x221l#overview
[84] http://www.bandrich.com/Data-Card_C500.html
[85] http://www.sierrawireless.com/productsandservices/AirPrime/Wireless_Modules/High-speed/MC7710.aspx
[86] http://www.sierrawireless.com/Support/Downloads/AirCard/USB_Modems/AirCard_320U.aspx
[87] http://www.yotamagaz.ru/lu150/
[88] http://www.yota-system.ru/oborudovanie-4g/modemy/new+modem/
[89] http://wifi.yota.ru/
[90] http://www.vodafone.com/content/dam/group/devices/downloads/specs/K4305specs.pdf
[91] http://www.ztedevices.com/product/data_card/2568654e-c0d9-479d-a77f-17c340ff211d.html
[92] http://routerboard.com/R52
[93] http://routerboard.com/R52H
[94] http://www.techniclan.com/Wireless_mPCI_TMP-5414A_11abg_108Mbps_Atheros_solution,p,74.html
[95] http://www.compex.com.sg/fullDescription.aspx?pID=23
[96] http://www.compex.com.sg/fullDescription.aspx?pID=25
[97] http://dbii.com/f20.html
[98] http://dbii.com/f50.html
[99] http://www.dbii.com/f20-PRO.html
[100] http://www.dbii.com/f50-PRO.html
[101] http://www.tp-link.com/en/products/details/?model=tl-wn751nd
[102] http://www.tp-link.com/en/products/details/?model=tl-wn851nd
[103] http://www.staros.com/documentation/KXS30SG%20Sell%20Sheet.pdf
[104] http://www.zcomax.cz/Xg650.aspx
[105] http://www.routerboard.com/prices.html
[106] http://routerboard.com/R5H
[107] http://store.dcsindo.com/interfaces/wireless-minipci/wmia-198n.html

[108] http://www.compex.com.sg/fullDescription.aspx?pID=96
[109] http://www.compex.com.sg/fullDescription.aspx?pID=32
[110] http://dbii.com/f52N-PRO.html

Bandwidth Managment and Queues

- Vlans on Mikrotik environment -- under construction
- NetworkPro on Quality of Service A word from the specialists
- QoS Workshop @ MUM USA 2011, pdf [1] by Janis Megis, MikroTik (see tiktube [2] for video)
- QoS Best Practice @ MUM USA 2009, pdf [3] by Janis Megis, MikroTik
- Transparent Traffic Shaper
- PCQ Examples
- Per-Traffic Load Balancing
- How to apply different limits for Local/Overseas traffic
- Different limits for Local/Overseas traffic for 3 bandwidth rates using PCQ and Queue Tree
- Queue with Masquerading and Internal Web-Proxy
- Queue Tree with more than two interfaces
- Limit Different Bandwidth In Day and Night
- Different bandwidth in day and night for several categories of users
- Hotspot, apply different limits and different traffic priorities
- Basic Internet Sharing with PCQ Bandwidth Limiting
- Basic Traffic Shaping Based on Layer-7 Protocols
- Traffic Priortization, RouterOS QoS Implemetation
- DSCP based QoS with HTB
- Bandwidth Control in a Service Provider network - under construction
- VoIP The long hidden VoIP article. VoIP prioritization example
- Bandwith control on ADSL link
- PCQ and Hotspots, and exempting upstream resources from rate limit
- Cache Hit flow control using PCQ

References

[1] http://mum.mikrotik.com/presentations/US11/us11-megis.pdf
[2] http://www.tiktube.com
[3] http://mum.mikrotik.com/presentations/US09/megis_qos.pdf

Firewall

Miscellaneous

- Basic universal firewall script
- Attempt To Detect And Block Bad Hosts
- Securing A New RouterOS Install
- Spam Filtering with Port Forwarding and Geo-Location
- Bridge Filter - Blocking DHCP Traffic
- Protecting your customers
- Securing your router
- How to secure a network using ARP
- Drop IM Using L7
- Drop port scanners
- Redirect mail traffic to a specified server
- How to Block Customer
- Dmitry on firewalling
- NetworkPro on firewalling - optimized, for two Public interfaces
- Forwarding a port to an internal IP
- More about N-th matcher
- How to autodetect infected or spammer users and temporary block the SMTP output
- Bruteforce login prevention (FTP & SSH)
- How to Block Websites & Stop Downloading Using Proxy
- L7 Filter
- Calea
- NTH in RouterOS 3.x
- Blocking specific sites with address lists
- NAT Tutorial
- Use host names in firewall rules
- DoS attack protection
- How PCC works (beginner)
- Hairpin NAT
- Port Knocking
- Reply Response Patterns
- Block Download Based Download Sizes
- Block Download Extention With Firewall Filter - mp3 , exe , ...
- How to Detect and Block Hotspot Shield program traffic(openvpn application)
- How to block non DHCP clients without the firewall
- How to Detect and Block UltraSurf program traffic
- How to Detect and Block TOR Browser traffic
- How To Block Facebook

Web-Proxy or External Cache Servers (SQUID, ISA, any Open Source Cache Server)

- Examples for Use Caching Server (5 Main Idea's)
- Squid3+TPROXY4+Mikrotik5
- open source caching server

Firewall mangle prerouting

- Live-IP-CONCEPT route a IP in any interface with Original ID
- Upload_wan_download_another

Firewall Scripts

- Home Firewall
- BOGON Address List

Monitoring

- SNMP Write
- RouterOS and Traffic-Flow
- How to make Ethereal/Wireshark to accept MikroTik sniffer TZSP stream
- SNMP Proxy (different routers monitor via SNMP)
- Reading SNMP via PHP
- Easiest way to monitor interface traffic via mrtg
- Remote Management of multiple bridged routers
- Monitoring Network thru SMS Alerts
- Monitoring Mikrotik with Munin
- OS X Lion as a syslog server

User/Routing

Routing

- Split horizon
- Dynamic Routing Concepts
- ECMP load balancing with masquerade
- NTH load balancing with masquerade
- NTH load balancing with masquerade (another approach)
- Basic Internet Connection Sharing (NAT)
- Connection Sharing in a Single MAC-Address Restricted Service Access
- Multiple gateway simple failover
- Bonding
- Load Balancing over Multiple Gateways
- ECMP Failover Script
- Routing Questions
- Suggested literature (Book review)
- Policy Routing in RouterOS 2.9.x
- Policy Routing in RouterOS 3.x
- Route Selection Algorithm in RouterOS
- MME wireless routing protocol
- MME command reference
- PCC exemptions
- Your Name In Trace Respons For Your Users
- Policy Base Routing
- Layer-2 routing for Mesh networks
- Routing local + international + unshaped traffic through 3 separate adsl accounts
- Routing testing scenarious (Configuration examples made by MikroTik)
- Simple Static Routes Example

Bridge

- Transparent Bridge in non-wireless

MPLS

See MPLS.

BGP

- BGP HowTo & FAQ
- BGP Best Path Selection Algorithm
- BGP Case Studies
- MT BGP configuration with corresponding Cisco settings
- BGP soft reconfiguration alternatives in RouterOS
- Using scope and target-scope attributes
- Limiting maximum number of prefixes accepted
- BGP nexthop selection and validation in RouterOS 3.x

- BGP Load Balancing with two interfaces

OSPF

- Steps of making neighborship between OSPF routers
- OSPF and Point-to-Point interfaces
- OSPF and Area summaries
- OSPF to simulate full duplex links with redundancy
- OSPF and PPPoE Setup
- OSPFv3 with Quagga
- OSPF summarization and redistribution complex example
- Mutual internet backup between two small ISP

IPv6

- Creating loopback interface for IPv6
- Setting up an IPv6 tunnel via a tunnel broker

RIP

- Routing Information Protocol Concept
- How to set wireless client and Ethernet

Multicast

- Multicast
- IGMP-Proxy
- Multicast Routing in RouterOS 3.x
- Multicast SPT Switchover

Scripts

Setup

- How to Make an Automated Configuration and Uninstall
- A script to set up WAN/LAN/WLAN to get you started

General

- Traffic Prioritization Script
- Automated Billing Script
- Automated Usage Script without usermanager
- Dynamic DNS Update Script for ChangeIP.com
- Dynamic DNS Update Script for ChangeIP behind NAT
- Dynamic DNS Update Script for EveryDNS
- Dynamic DNS Update Script for dynDNS
- Dynamic DNS Update Script for dynDNS behind NAT
- Dynamic DNS Update Script for DNSoMatic.com
- Dynamic DNS Update Script for DNSoMatic.com behind NAT
- Dynamic DNS Update Script for Hurricane Electric DNS
- Dynamic DNS Update Script for No-IP DNS
- Email setup/troubleshooting
- Hurricane Electric IPv6 Tunnel - IPv4 Endpoint updater
- Using 'find' command to filter a command output
- GPS text file converter to Google Earth/Maps
- Remove BUSY status DHCP Leases to solve malfunction of DHCP server
- Scheduled disconnect for WAN-Interface e.g. DSL
- Scheduled check for loaded interfaces (auto adding queue to some IP or interface)
- Sending text out over a serial port
- Set global and local variables
- Setting static DNS record for each DHCP lease
- Sending your self an e-mail with DSL interface IP address
- Queue tree and e-mailing stats
- How to control shared users when PPP server is used with Radius
- Script to monitor unexpected script failure
- A Bit of Sounds
- Use host names in firewall rules
- Script to find the day of the week
- Calculate with decimal numbers
- Use Functions in CMD Script
- Script to create directory
- Backup graphing data
- Calea perl trafr
- IP Pool Statistics
- Log Parser - Event Trigger Script
- Super Mario Theme
- Routing via a DHCP allocated gateway (when this address could change and is not a default route)

- Get active VPN connections via e-mail (PPTP and L2TP)
- Get active VPNs, connected wireless stations, active Hotspot sessions and connected administrators via e-mail
- Using scripting to overcome the inability to specify number ranges on the command line
- Converting network and gateway from routing table to hexadecimal string
- Useful Bash Scripts
- Update static DNS entries every 10mins. (Specifically in cases where the upstream ISP "loadbalance" between SMTP servers by using a low TTL on their SMTP DNS)
- Use Mikrotik as Fail2ban firewall
- Cool Console

Hotspot

- Reset Hotspot user count
- Enable/Disable new guest user account daily
- PayPal with hotspot and walled garden bypass
- Expire users a after number of days
- Add a data limit to trial hotspot users

Modifying Router Settings 'on the fly'

- Enable and Disable P2P connections
- Generate bogons firewall chain based on routing-marks
- Limiting a user to a given amount of traffic (using firewall)
- Limiting a user to a given amount of traffic II (using queues)
- Limiting a user to a given amount of traffic with user levels (using queues)
- Limit Different Bandwidth In Day and Night
- Enable Disable Firewall Rules
- Blocking Rapidshare.com web page
- Random MAC/Ethernet address generate and apply
- Using Fetch and Scripting to add IP Address Lists

Resilience/Monitoring

- Monitoring Script
- ECMP Failover Script
- Improved Netwatch
- Improved Netwatch II
- Failover con Netwatch III
- Failover via Netwatch III (English)
- Force Disconnect Wireless Stations with Low CCQ
- Monitor logs, send email alert / run script
- PPP Keepalive ping
- Send email about reboot
- Easy Failover using only a script
- Secure L2TP server for IPSec clients only

System Maintenance

- BackupROS (Centralized Backups) - by Nahuel Ramos (new!)
- Centralized Automated Backups via Email with Procmail and Perl
- Automatic Backup with Centralized Storage
- Antenna Alignment with RB532 LED
- Audible signal test
- Logging SNR and thruput values
- Logging Average CCQ and Wireless Clients Stats
- Generate routes for stress testing BGP functionality
- Improved Semi-automatic system-update script
- Scheduled sending of an email with system backup attached
- Flash Friendly Backup Script
- Semi-automatic system-update by script
- Use SSH to execute commands (DSA key login)
- Auto upgrade script V3.x
- sending mails when on battery or battery low
- Delete ARP trafic for arp table
- Add Static DHCP Leases to ARP List
- Batch deployment of DSA key (SSH) and schedule backup with export
- Automated Upgrade/Downgrade script V3.9+
- Improved auto upgrade script v3.X
- Remotely change password for managers
- Monitor input voltage on RB333/433AH
- Reboot Boards due to low Memory with notification
- Yet Another Alignment Script With LEDs And Sound
- Alignment Script that "reads back" RSSI with beeps
- Netwatch on web
- Sync Address List with DNS Cache
- Sync Address List from DNS Lookup Results - CNAME and A Records
- SXT 5HnD Alignment Script
- Semi-Automating CPE ROS/Firmware/script updates and setting changes

Reporting

- Firewall Usage
- Automatically_Create_Simple_Queues

Tunnels

- PPtP Server / VPN
- PPtP Client / VPN
- PPTP VPN - multiple ADSL remote locations to Cental Office
- IPSec VPN with Dynamic Routing / Mikrotik and Cisco
- IPSec VPN / Mikrotik and Linksys BEFVP41
- VPN with Virtual Routing and Forwarding / Mikrotik and Cisco
- OpenVPN
- Layer2 VPN Server
- MikroTik RouterOS and Windows XP IPSec/L2TP
- IPSec VPN between MikroTik RouterOS and SonicWall SonicOS Enhanced
- PPPoe Server / VPN
- MikroTik router to CISCO PIX Firewall IPSEC
- Routing through remote network over IPsec
- L2TP + IPSEC between 2 Mikrotik routers
- VPN (any type) between 2 Mikrotik routers and no static IP addresses
- L2TP + IPSEC between Mikrotik router and a PC
- IPSEC between Mikrotik router and a Shrew_client
- OpenVPN Configuration Step by Step
- SSTP step-by-step

Wireless Setups

- Making a simple wireless AP
- 802.11n Setup guide for RouterOS v4.03beta
- Dual Link Setup with OSPF
- Dual Link using two radios and OSPF
- Transparently Bridge two Networks with WDS
- Transparently Bridge two Networks with EoIP
- Transparently Bridge two Networks using MPLS/VPLS
- Transparently Bridge two Networks using two VPLS tunnels
- Forget Bandwidth control, Add a second DHCP router on a single Wireless link!
- How to create a transparent AP with more than 1 wireless cards
- Wireless repeater
- Association establishment rules on AP
- Access and bandwidth limitation
- Mesh WDS setup with RSTP
- Mesh wireless; HWMP+
- Antenna Alignment with RB532 LED
- Link possibility calculator (beta version) [1]
- CPE alignment method
- Nstreme dual Step-by-Step
- Fix Broken Wireless Following an Upgrade
- 802.1q Trunk extension over Wireless P2P Link
- PTP Links - A Step By Step Guide

- Bridge Network With Wireless Modes
- Wireless WDS Mesh
- Connect to an Available Wireless Network

References

[1] http://www.mikrotik.com/test_link.php

Manual:MPLS

Sub Categories

List of reference sub-pages

- **Interface**
  - vpls
  - traffic-eng
- **MPLS**
  - ldp
  - traffic-eng

Case studies

General

- MPLS Overview and RouterOS MPLS Implementation Status
- EXP bit behaviour
- L2MTU

Layer2 VPN

- LDP and LDP based VPLS
- BGP based VPLS
- Cisco style VPLS
- VPLS Control Word

Layer3 VPN

- Virtual Routing and Forwarding (VRF)
- OSPF as PE-CE routing protocol
- EBGP as PE-CE routing protocol

Traffic Engineering

- TE Tunnels
- TE Tunnel Bandwidth Control

List of examples

General

- MPLS over PPPoE

Layer2 VPN

- P2P L2VPN to Juniper router

Layer3 VPN

- A complete Layer-3 MPLS VPN example
- VRF Route Leaking
- Internet access from VRF
- Internet access from VRF with NAT

Traffic Engineering

- Simple TE configuration
- TE tunnels for VPLS

Summary

MikroTik RouterOS [1] **supports MPLS.** All MikroTik RouterBOARD [2] hardware products support MPLS.

General Porperties

Property	Description
dynamic-label-range (*range of integer[16..1048575]*; Default: **16-1048575**)	Range of Label numbers used for dynamic allocation. First 16 labels are reserved for special purposes (as defined in RFC). If you intend to configure labels statically then adjust dynamic default range not to include numbers that will be used in static configuration.
propagate-ttl (*yes* | *no*; Default: **yes**)	Whether to copy TTL values from IP header to MPLS header. If this option is set to **no** then hops inside MPLS cloud will be invisible from traceroutes.

Forwarding Table

Sub-menu: `/mpls forwarding-table`

Entries in this sub-menu shows label bindings for specific routes that will be used in MPLS label switching. Properties in this menu are read-only

Property	Description
bytes (*integer*)	Total number of packet bytes matched by this entry
destination (*IP/Mask*)	Destination prefix for which labels are assigned
in-label (*integer*)	Label number for incoming packet
interface (*string*)	
ldp (*yes* | *no*)	Whether labels are LDP signaled
nexthop (*IP*)	IP address of the nexthop
out-label (*integer*)	Label number which is added or switched to for outgoing packet.
packets (*integer*)	Number of packets matched by this entry
traffic-eng (*yes* | *no*)	Shows whether entry is signaled by RSVP-TE (Traffic Engineering)
vpls (*yes* | *no*)	Shows whether entry is used for VPLS tunnels.

For example we have forwarding table as shown below.

```
[admin@RB493G] /mpls forwarding-table> print
Flags: L - ldp, V - vpls, T - traffic-eng
 #    IN-LABEL      OUT-LABELS    DESTINATION                    IN NEXTHOP
 0    expl-null
 1 L 105                          10.255.255.36/32               lo 10.5.101.36
 2 L 120           112            3.3.3.1/32                     lo 10.5.101.3
 3 L 121           113            3.3.3.2/32                     lo 10.5.101.3
[admin@RB493G] /mpls forwarding-table>
```

You can see that all labels are LDP signaled. Note that for entry #1 there is no out-label, it means that MPLS label switching will not occur, packet will be sent out as regular IP packet. In the other hand entry #2 has in-label and out-label, which means that during packet forwarding label will be switched from 120 to 112.

Interface

Sub-menu: `/mpls interface`

This menu allows to configure MTUs including MPLS headers that interface can forward without fragmentation.

Note: If Ethernet card does not support Jumbo frames, then MPLS MTU for all interfaces on all devices participating in LSP should be set to 1500

Properties

Property	Description
comment (*string*; Default:)	Short description of the interface
disabled (*yes* | *no*; Default: **no**)	If set to **yes** then this configuration is ignored.
interface (*string* | *all*; Default: **all**)	Interface name to which apply settings. If set to **all** then the same config will be used for every interface if there is no specific configuration for the interface.
mpls-mtu (*integer [512..65535]*; Default: **1508**)	Option represents how big packets can be carried over the interface with added MPLS labels. Read More >>

In RouterOS by default have entry which sets MS MTU to 1508 for all interfaces.

```
[admin@RB493G] /mpls interface> print
Flags: X - disabled
 #   INTERFACE                                                       MPLS-MTU
 0   all                                                                 1508
```

Local Bindings

Sub-menu: /mpls local-bindings

This sub-menu shows labels bound to the routes locally in the router. In this menu also static bindings can be configured if there is no intention to use any of dynamic protocols (like LDP).

Properties

Property	Description
comments (*string*; Default:)	Short description of the entry
disabled (*yes* | *no*; Default: **no**)	
dst-address (*IP/Mask*; Default:)	Destination prefix for which label is assigned
label (*integer[0..1048576]* | *alert* | *expl-null* | *expl-null6* | *impl-null* | *none*; Default:)	Label number assigned to destination.

Read-only Properties

Property	Description
adv-path ()	
advertised (*yes* | *no*)	Whether binding was advertised to the neigbors
dynamic (*yes* | *no*)	Whether entry was dynamically added
egress (*yes* | *no*)	
gateway-route (*yes* | *no*)	Whether destination is reachable through the gateway.
local-route (*yes* | *no*)	Whether destination is locally reachable on the router
peers (*IP:label_space*)	IP address and label space of the peer to which this entry was advertised.

Remote Bindings

Sub-menu: `/mpls remote-bindings`

Sub-menu shows label bindings for routes received from other routers. This table is used to build Forwarding Table

[Top | Back to Content **]**

References

[1] http://mikrotik.com/software.html
[2] http://routerboard.com

Manual:Virtualization

Applies to RouterOS: 3, v4

RouterOS has three different Virtualization implementations. Choose your topic:

- Metarouter
- Xen
- Kvm

Metarouter

Metarouter is created by MikroTik and currently is supported only on RouterBOARD 4xx series (mips-be) and RB1000 series (powerpc). Currently Metarouter can only create RouterOS virtual machines.

We are planning to add more features to Metarouter, so that it will even exceed Xen in functionality. New hardware support will also be added to Metarouter

Xen

Xen is based on the Linux Xen Virtual machine project, and current RouterOS implementation is supported only on RouterOS X86 systems (PCs). Xen can create Virtual machines of different Operating Systems that supports Xen.

Kvm

Kvm is based on Linux Kvm virtualization software and requires your CPU to support virtualization. Kvm is available only on x86 systems.

Usage Examples

The following are just a few of possible scenarios where virtual machines could be used (some of these currently are possible only in Xen, but Metarouter features will be expanded to allow even more functionality):

In the datacenter

- consolidate a number of routers on one hardware platform
- consolidate routing services and higher levels services such a VOIP switches in the same box
- use a guest machine on top of a router for custom features such as accounting, LDAP or legacy networking
- redundant routers much easier and cheaper to have available in case of crashed systems

In the hosting center

- use RouterOS and extensive networking features as a host with a server (mail, http, ftp...) running as guest or multiple guest virtual machines

- offer virtual routers with VPN solutions that give a network administrator customer his own router on a highspeed backbone to make any kind of tunneled intranet or simply VPN access system

At the wireless ISP client site

- set up two isolated routers and set the wireless control only for the router controlled by the WISP while the Ethernet side router is fully under the clients control

At multiclient sites (such as office buildings)

- in locations serving multiple clients by Ethernet from one backbone connection (wired or wireless), give each customer control over his own isolated virtual router

For network planning and testing

- build a virtual network on one box with the same topography as a planned network and test the configurations so that the fine tuning of the configurations can be done in the lab and not in the field, simulate and monitor the network with advanced scripting and The Dude network monitor utility

In custom applications

- develop your own programs (and even Linux distributions) that can be installed on MikroTik supported platforms with minimum difficulty as software patches and virtual drivers are provided for guest systems
- use low cost RouterBOARD embedded systems easily with your own Linux and the advantage that it will work across all RouterBOARDS with the same CPU

Use Metarouter to Implement Tor Anonymity Software

This article describes the steps to set up Tor Anonymity software behind a Mikrotik Hotspot using a Metarouter instance. The Metarouter image presented here is for RB4xx MIPSBE boards that support OpenWRT Metarouter images.

The end result is a NATed network that routes only encrypted tor traffic for end users. The only ports that are open to end users include 80 tcp, 53 udp, 8118 tcp and 9050 tcp. 8118 tcp is the Privoxy proxy which acts as a standard http proxy to the Tor Socks proxy. Provoxy also has enhanced privacy features such as "removing ads and other obnoxious junk" [1]. Port 9050 tcp is the Socks proxy available for routing traffic through the tor network. All other ports are blocked for security and anonymity reasons. This configuration may be used to set up a wifi network which automatically routes traffic through tor.

Tor Background

What is Tor?

Put simply, Tor is anonymity software that protects a source computer from eavesdropping by a third party. Tor routes internet packets through a series of encrypted proxies. Each proxy in the chain knows a part of the request, but not the entire request. The destination server also does not know what the source is. Tor may also be referred to as Onion routing. Tor is an open source project run by volunteers from around the world.

From the Tor web site [2]

> "Tor is a network of virtual tunnels that allows people and groups to improve their privacy and security on the Internet. It also enables software developers to create new communication tools with built-in privacy features. Tor provides the foundation for a range of applications that allow organizations and individuals to share information over public networks without compromising their privacy."

Why use Tor?

From the Tor web site [2]

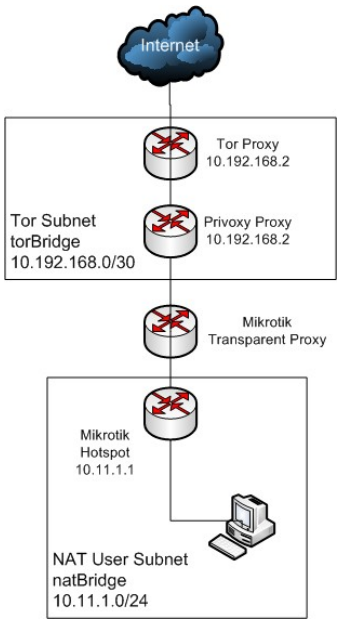
> "Using Tor protects you against a common form of Internet surveillance known as "traffic analysis." Traffic analysis can be used to infer who is talking to whom over a public network. Knowing the source and destination of your Internet traffic allows others to track your behavior and interests. This can impact your checkbook if, for example, an e-commerce site uses price discrimination based on your country or institution of origin. It can even threaten your job and physical safety by revealing who and where you are. For example, if you're traveling abroad and you connect to your employer's computers to check or send mail, you can inadvertently reveal your national origin and professional affiliation to anyone observing the network, even if the connection is encrypted."

Tor Web Site

More information about The Tor Project is available at available at The Tor Homepage [3].

Network Description

The network design requires that users be behind a NAT connection. The metarouter runs the Tor service and all web traffic is routed through it. By design, to protect user privacy, only port 80 tcp, port 53 udp, 8118 tcp (privoxy proxy) and port 9050 tcp (tor socks proxy) are open to users.



Mikrotik Network Configuration

Set up bridges

```
/interface bridge
add name=torBridge
add name=natBridge

/interface bridge port
add interface=ether2 bridge=natBridge
add interface=ether3 bridge=natBridge
add interface=wlan1 bridge=natBridge
```

These commands set up the necessary bridges and add interfaces to the natBridge. In this example, an RB433AH with wifi card is being used. Three physical ports will be added to the natBridge (ether2, ether3 and wlan1). Ether1 is the port for the internet connection.

Configure Wifi AP

```
/interface wireless set [find name="wlan1"] disabled=no \
mode=ap-bridge band=2.4ghz-b/g frequency=2412 ssid="Tor Anonymous Web"
```

This command configures wlan1 interface SSID, mode, band and channel. Settings such as wifi encryption may be adjusted as desired.

Add IP addresses

```
/ip address add interface=ether1 address=192.168.3.254/24 disabled=no
/ip address add interface=natBridge address=10.11.1.1/24 disabled=no
/ip address add interface=torBridge address=10.192.168.1/30 disabled=no
```

Ether1 is the internet IP address. In this example, 192.168.3.0/24 network is being used.

Configure default route (if needed)

```
/ip route add dst-address=0.0.0.0/0 gateway=192.168.3.7
```

Configure DHCP server for natBridge

```
/ip pool add name="nat-DHCP" ranges="10.11.1.10-10.11.1.250"
/ip dhcp-server network add address=10.11.1.0/24 gateway=10.11.1.1 dns-server=10.192.168.2
/ip dhcp-server add interface="natBridge" lease-time="1:00:00" name="nat-DHCP-Server" \
  address-pool="nat-DHCP" authoritative=yes disabled=no
```

Firewall NAT rules

```
/ip firewall nat
# only masquerade torBridge
add chain=srcnat action=masquerade src-address=10.192.168.0/30 disabled=no

# transparent proxy redirect
add chain=dstnat in-interface=natBridge protocol=tcp dst-port=80 \
  action=redirect to-ports=8080 disabled=no

# DNS, privoxy and Tor socks forward rules for natBridge
add chain=dstnat in-interface=natBridge protocol=udp dst-port=53 \
  action=dst-nat to-addresses=10.192.168.2 to-ports=53 disabled=no
add chain=dstnat in-interface=natBridge protocol=tcp dst-port=8118 \
  action=dst-nat to-addresses=10.192.168.2 to-ports=8118 disabled=no
add chain=dstnat in-interface=natBridge protocol=tcp dst-port=9050 \
  action=dst-nat to-addresses=10.192.168.2 to-ports=9050 disabled=no

# DNS, privoxy and Tor socks forward rules for ether1 (optional)
add chain=dstnat in-interface=ether1 protocol=udp dst-port=53 \
  action=dst-nat to-addresses=10.192.168.2 to-ports=53 disabled=no
```

```
add chain=dstnat in-interface=ether1 protocol=tcp dst-port=8118 \
  action=dst-nat to-addresses=10.192.168.2 to-ports=8118 disabled=no
add chain=dstnat in-interface=ether1 protocol=tcp dst-port=9050 \
  action=dst-nat to-addresses=10.192.168.2 to-ports=9050 disabled=no
```

In this configuration, we don't want to masquerade the natBridge directly. Instead, in order to maintain anonymity, privacy and encryption, only torBridge is masqueraded. Users may only use port 80 tcp and 53 udp by default. Ports 9050 (Tor socks proxy) and 8118 (Privoxy http proxy) are also available in order for users to configure other services such as https or messaging. These nat rules also redirect all port 80 requests to Mikrotik transparent proxy.

Configure Mikrotik Transparent Proxy

```
/ip proxy set enabled=yes parent-proxy=10.192.168.2 parent-proxy-port=8118 \
  cache-on-disk=no max-fresh-time=1h
```

Configure Hotspot (optional)

```
/ip hotspot
add name="Tor" address-pool=nat-DHCP interface=natBridge idle-timeout=20m disabled=no
/ip hotspot user profile
set default keepalive-timeout=5m shared-users=1000 transparent-proxy=yes \
  rate-limit=512k/1024k
/ip hotspot user
  add comment="" disabled=no name=tor password=tor profile=default
/ip hotspot walled-garden
add action=allow comment="" disabled=no dst-host=*.torproject.org
add action=allow comment="" disabled=no dst-host=*.eff.org
/ip dns
set servers=10.192.168.2
```

These commands are optional and will set up hotspot for Tor access with username tor password tor and bandwidth limiting set to 512kbps down and 1024kbps up. Hotspot login page files with a standard accept button are avilable here [4]. Also, if DNS server is not already configured, it should be set at this time.

Mikrotik Metarouter Configuration

Obtain Tor Metarouter image via download link

Download the metarouter image from the download link [5] and upload the image to the router's root directory.

Import Metarouter image

```
/metarouter import-image memory-size=32 file-name=openwrt-22250-tor-image.tar.gz
```

After uploading the .tar.gz file to the root directory, this command will import and start the metarouter image.

Configure Metarouter name and network interface

```
/metarouter set 0 name=tor
/metarouter interface add type=dynamic dynamic-bridge=torBridge virtual-machine=tor
```

The first command names the new Metarouter virtual machine. The second command sets up a dynamic interface for the metarouter the torBridge interface.

Set up scheduler to periodically reboot metarouter

```
/system scheduler add disabled=no interval=6h name=restartTor \
on-event="/metarouter set [find name=\"tor\"] \
disabled=no\r\
\n:delay 5\r\
\n/metarouter set [find name=\"tor\"] disabled=no" policy=\
reboot,read,write,policy,test,password,sniff,sensitive
```

Metarouter needs to be restarted periodically in order for the Tor image to run smoothly.

Set up Tor Relay or Bridge (optional)

The Tor Network relies on the existence of Tor relays, bridges and exit nodes. Anyone may run a relay or bridge and the Tor web site encourages this. It is also possible to run an exit node, however doing this is outside the scope of this article. More information about relays, bridges and exit nodes is available at the Tor Project web site.

Mikrotik Port Forward For Tor Bridge

If Tor bridge is desired, port 443 tcp needs to be reachable from the external network. In RouterOS:

```
/ip firewall nat
add chain=dstnat in-interface=ether1 protocol=tcp dst-port=443 \
  action=dst-nat to-addresses=10.192.168.2 to-ports=443 disabled=no
```

Mikrotik Port Forward For Tor Relay

If Tor relay is desired, port 9001 tcp needs to be reachable from the external network. In RouterOS:

```
/ip firewall nat
add chain=dstnat in-interface=ether1 protocol=tcp dst-port=9001 \
  action=dst-nat to-addresses=10.192.168.2 to-ports=9001 disabled=no
```

Metarouter console configuration

The next step is to configure Tor in the OpenWRT metarouter. There are several pre-written Tor configuration files in /etc/tor. To run a bridge or relay, copy the relevant file to the running configuration and restart Tor as in the following example. torrc.bridge is the bridge configuration, torrc.relay is the relay configuration and torrc.client is the client-only configuration. By default, the torrc.client configuration is enabled.

```
root@OpenWrt:/# cd /etc/tor
root@OpenWrt:/etc/tor# ls -l
-rw-r--r--    1 root     root          7141 Aug 18 00:44 torrc
-rw-r--r--    1 500      500           7219 Aug 18 00:43 torrc.bridge
-rw-r--r--    1 500      500           7143 Aug  8 00:49 torrc.client
-rw-r--r--    1 500      500           7141 Aug  8 02:16 torrc.relay
root@OpenWrt:/etc/tor# cp torrc.relay torrc
root@OpenWrt:/etc/tor# /etc/init.d/tor stop
root@OpenWrt:/etc/tor# /etc/init.d/tor start
Aug 18 00:45:18.889 [notice] Tor v0.2.1.26. This is experimental software. Do no
```

```
t rely on it for strong anonymity. (Running on Linux mips)
Aug 18 00:45:18.923 [notice] Choosing default nickname 'openwrt'
Aug 18 00:45:18.925 [notice] Your ContactInfo config option is not set. Please c
onsider setting it, so we can contact you if your server is misconfigured or som
ething else goes wrong.
Aug 18 00:45:18.954 [notice] Initialized libevent version 1.4.13-stable using me
thod epoll. Good.
Aug 18 00:45:18.960 [notice] Opening OR listener on 0.0.0.0:9001
Aug 18 00:45:18.964 [notice] Opening Socks listener on 10.192.168.2:9050
Aug 18 00:45:18.966 [notice] Opening DNS listener on 10.192.168.2:53
root@OpenWrt:/etc/tor#
```

Occasionally, errors will be displayed when restarting Tor. This is because sometimes Tor does not die as it should when the stop command is issued. If errors are displayed, try the /etc/init.d/tor stop command a few more times until tor is able to be started by a tor start command. Alternatively, the metarouter may be rebooted to get everything set up properly. Tor logs are available in /var/log/tor/notices.log.

See Also

- Tor Home Page [6]
- Privoxy Home Page [1]
- /etc/tor/torrc Manual [7]
- /etc/privoxy/config Manual [8]
- Tor Volunteer Opportunities [9]
- Firefox Tor Button (Enable/disable Tor with a click) [10]
- Tor Use Cases [11]
- Video: Why Tor is slow and what is being done about it [12]

References

[1] http://www.privoxy.org/
[2] https://www.torproject.org/overview.html.en
[3] http://www.torproject.org
[4] http://webasdf.dyndns.org/tor/hotspotLoginFiles.tar.gz
[5] http://www.webasdf.com/tor/openwrt-22250-tor-image.tar.gz
[6] https://www.torproject.org/
[7] http://www.torproject.org/tor-manual.html.en
[8] http://www.privoxy.org/user-manual/config.html
[9] https://www.torproject.org/volunteer.html.en
[10] http://www.torproject.org/torbutton/
[11] https://www.torproject.org/torusers.html.en
[12] http://www.youtube.com/watch?v=J-7iNS0VzGU

User/IPv6

- Overview and examples
- Setting up an IPv6 tunnel via a tunnel broker
- Creating loopback address for IPv6
- OSPFv3 with Quagga
- Setting up an IPv6 tunnel via 6to4
- Setting up DHCPv6

User Management

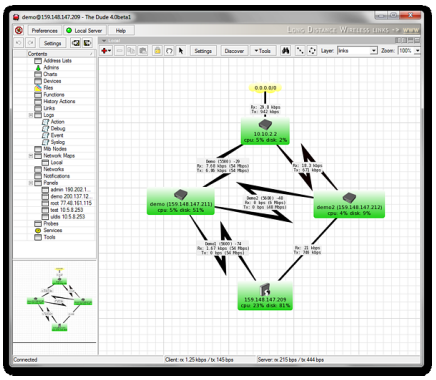
- HotSpot Redirect to external login page
- HotSpot external login page
- Pppoe_server_with_profiles
- Hotspot_server_setup
- SSL Certificate setup
- Manual HotSpot Setup (In Greek)
- Troubleshooting HotSpot
- PPTP_Server_With_Profile
- Notify your customers internet is down, monitor connectivity
- Free Internet access through Hotspot when RADIUS is down
- Payment Reminders
- Software HotSpot

The Dude

The Dude [1] is a free application by MikroTik, which can dramatically improve the way you manage your network environment.

It will automatically scan all devices within specified subnets, draw and layout a map of your networks, monitor services of your devices and execute actions based on device state changes.

Not only can you monitor your devices, you can also manage them. Mass upgrade RouterOS [2] devices, configure them right from within the Dude interface, run network monitoring tools etc.



The Dude documentation

General usage

- Installation and requirements
- First launch of the Dude
- Main window overview
- Search and Export to PDF/CSV
- Web interface

Devices

- Device list
  - Upgrading RouterOS with the Dude
- Device map
- Discovering devices
- Adding and Editing devices
- Links
- Networks

Monitoring

- Services and Outages
- Notifications
- Charts
- Functions
- Agents / Dude on RouterOS
- Logs
- Syslog server
- MIB nodes
- Probes

Settings

- Server settings
- Server files
- Admins
- Address lists
- History
- Panels
- Tools

Misc Documents

- Version changelog
- The Dude License
- Dude 4 documentation PDF [3] *Note*: the same document can be made by choosing **Create Book** at the top of this page.

User articles

How-To's:

- Dude interface translations
- Dude windows installation
- Dude Linux Installation
- The Dude/Dude as a Linux Service
- Exporting and Importing Configuration
- Before doing anything guide
- Getting started with Functions and probes
- Quick guide to a good probe
- Using Discovery
- Device management

- Graphing Client Signal Strength
- Managing Multiple Remote Bridged Routers
- Dude as Syslog Server
- Dude como Servidor Syslog (el español)
- Custom probe settings
- Extra Tools
- Start The Dude with Shortcut
- Email notifications
- Email notifications using... Gmail
- Alternate SMTP port for notifications
- View and Graph the Number of Wireless Clients
- Display voltage for Mipsbe devices
- DOCSIS Statistics (Arris C3, Motorola cablemodems) (new)

References

[1] http://www.mikrotik.com/thedude.php
[2] http://www.mikrotik.com/software.html
[3] http://mikrotik.com/pdf/dude4.pdf

User Manager

Introduction

- What is User Manager
- Requirements
- Supported browsers
- Demo
- Differences between version 3 and version 4-test

Getting started

- Download
- Install
- Create first subscriber
- First log on User Manager web

Quick start

- User Manager and HotSpot
- User Manager and PPP servers
- User Manager and DHCP
- User Manager and Wireless
- User Manager and RouterOS user

Concepts explained

Common

- Customers
- Users
- Routers
- Sessions
- Payments
- Reports
- Logs
- Customer permission levels
- Character constants
- Active sessions
- Active users
- Customer public ID

Version 4.x test package specific

- Profiles
- Limitations
- User data templates
- MAC binding
- Languages
- CoA (Radius incoming)

Version 3.x specific

- Subscribers
- Credits
- User prefix
- Time, traffic amount and rate limiting
- Prepaid and unlimited users
- Voucher template

Reference

Web interface

- Search patterns
- Tables:
  - Sorting
  - Filtering
  - Division in pages
  - Multiple object selection
  - Operations with selected objects
  - Minimization
  - Links to detail form
- Detail forms
- Page printing

Customer page

- Setup
- How to find it?
- Sections
  - Status
  - Routers
  - Credits
  - Users
  - Sessions
  - Customers
  - Reports
  - Logs

User page

- Setup
- How to find it?
- Link to user page
- Sections
  - Status
  - Payments
  - Settings

User sign-up

- Setup
- Sign-up steps
  - Creating account
  - Activating account
  - Login

User payments

- Authorize.Net
- PayPal

Questions and answers

- Quick introduction into User Manager setup
- How to separate users among customers?
- How to create a link to user page?
- How to create a link to user sign-up page?
- Visual bugs since upgrade
- Cannot log in User Manager
- Too many active sessions shown
- What does "active sessions" refer to?
- How to make Hotspot and User Manager on the same router?
- How to make MAC authentication in the User Manager?
- How to turn off logging for specific Routers?

- How to create timed Voucher?
- Cannot access User Manager WEB interface
- Incorrect time shown for sessions and credits
- User Manager does not allow to login due to expired uptime
- How to debug PayPal payments
- How to send logs to a remote host, using SysLog

API PHP package

Client

The examples on this page use the PEAR2_Net_RouterOS [1] package. You can install it with Pyrus, PEAR, Composer or just *download the ".phar" file and include it from your PHP file*.

NOTE: Despite the name, PEAR(2) itself is NOT required.

NOTE: The client requires PHP 5.3.0 or later.

NOTE: The client should, in theory, work without any problems for large replies and commands, as well as any and all RouterBOARD devices, but has not been extensively tested with such. Please report any such experiences (positive or negative ones) at the forums [2].

Credits and legal stuff

Author: Vasil Rangelov, a.k.a. boen_robot (boen [dot] robot [at] gmail [dot] com)

License: LGPL 2.1 [3]

(Summary: Use the library as you like, no requirements or restrictions; If you modify the library and publish an application using that library, also publish the modified library itself with the original credits preserved and under the same license)

Examples

All examples assume that you used Pyrus or PEAR for installation and have installed PEAR2_Autoload. Also, the router is assumed to be accessible with a local IP to the device PHP runs from. The client itself could work without these restrictions - they are specified here for clarity and consistency.

NOTE: You should be able to replace "PEAR2/Autoload.php" with the path to the ".phar" file, and have everything "just work".

Print router logs

The following example shows the router's log into a table. You should make sure this is not publicly visible, as it may give potential attackers useful info (especially the parts about a username having logged in by a particular protocol).

```
<?php
use PEAR2\Net\RouterOS;
require_once 'PEAR2/Autoload.php';

?><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
```

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <meta http-equiv="Content-Type" content="text/html;charset=UTF-8" />
        <title>RouterOS log</title>
        <style type="text/css">
            table, td, th {border: 1px solid black;}
            td span {outline: 1px dotted black;}
        </style>
    </head>
    <body>
        <?php
        try {
            $client = new RouterOS\Client('192.168.0.1', 'admin', 'password');
        } catch (Exception $e) {
        ?><div>Unable to connect to RouterOS.</div>
    <?php
        }

        if (isset($client)) {
        ?><table>
            <thead>
                <tr>
                    <th>Time</th>
                    <th>Topics</th>
                    <th>Message</th>
                </tr>
            </thead>
            <tbody><?php
                $logEntries = $client->sendSync(
                    new RouterOS\Request('/log print')
                )->getAllOfType(RouterOS\Response::TYPE_DATA);
                foreach ($logEntries as $entry) {
                    ?>

                <tr>
                    <td><?php echo $entry('time'); ?></td>
                    <td><?php
                    $topics = explode(',', $entry('topics'));
                    foreach ($topics as $topic) {
                        ?>

                        <span><?php echo $topic; ?></span><?php
                    }
                    ?>

                    </td>
```

```
                    <td><?php echo $entry('message'); ?></td>
                </tr><?php } ?>

            </tbody>
        </table>
    <?php } ?></body>
</html>
```

Ping from router

This example is particularly useful when you want to ping someone from *inside* the network while browsing the page from *outside* the network.

```
<?php
use PEAR2\Net\RouterOS;
require_once 'PEAR2/Autoload.php';

if (isset($_GET['act'])) {//This is merely to ensure the form was submitted.

    //Adjust RouterOS IP, username and password accordingly.
    $client = new RouterOS\Client('192.168.0.1', 'admin', 'password');

    //This is just one approach that allows you to create a multi purpose form,
    //with ping being just one action.
    if ($_GET['act'] === 'Ping' && isset($_GET['address'])) {
        //Ping can run for unlimited time, but for PHP,
        //we need to actually stop it at some point.
        $pingRequest = new RouterOS\Request('/ping count=3');
        $results = $client->sendSync($pingRequest->setArgument('address', $_GET['address']));
    }
}
?><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>Ping someone</title>
    </head>
    <body>
        <div>
            <form action="" method="get">
                <ul>
                    <li>
                        <label for="address">Address:</label>
                        <input type="text" id="address" name="address" value="<?php
                            if (isset($_GET['address'])) {
                                echo htmlspecialchars($_GET['address']);
                            }
                        ?>" />
```

```
                    </li>
                    <li>
                        <input type="submit" id="act" name="act" value="Ping" />
                    </li>
                </ul>
            </form>
        </div>
        <?php
if (isset($_GET['act'])) {//There's no need to execute this if the form was not submitted yet.
    echo '<div>Results:<ul>';
    foreach ($results as $result) {
        //Add whatever you want displayed in this section.
        echo '<li>Time:', $result->getArgument('time'), '</li>';
    }
    echo '</ul></div>';
}
        ?>
    </body>
</html>
```

"Change password" form for hotspot users

The script assumes you have already made a hotspot and do NOT make this file accessible in a walled garden, i.e. users must be logged in to access it. For convenience's sake, you may want to link to it from the status page.

```
<?php
use PEAR2\Net\RouterOS;
require_once 'PEAR2/Autoload.php';

$errors = array();

try {
    //Adjust RouterOS IP, username and password accordingly.
    $client = new RouterOS\Client('192.168.0.1', 'admin', 'password');

    $printRequest = new RouterOS\Request(
        '/ip hotspot active print',
        RouterOS\Query::where('address', $_SERVER['REMOTE_ADDR'])
    );
    $hotspotUsername = $client->sendSync($printRequest)->getArgument('user');
} catch(Exception $e) {
    $errors[] = $e->getMessage();
}

if (isset($_POST['password']) && isset($_POST['password2'])) {
    if ($_POST['password'] !== $_POST['password2']) {
        $errors[] = 'Passwords do not match.';
    } elseif (empty($errors)) {
```

```
        //Here's the fun part - actually changing the password
        $setRequest = new RouterOS\Request('/ip hotspot user set');
        $client($setRequest
            ->setArgument('numbers', $hotspotUsername)
            ->setArgument('password', $_POST['password'])
        );
    }
}

?><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>Change your hotspot password</title>
        <style type="text/css">
            #errors {background-color:darkred;color:white;}
            #success {background-color:darkgreen:color:white;}
        </style>
    </head>
    <body>
        <div>
            <?php if (!isset($hotspotUsername)) { ?>
                <h1>We're sorry, but we can't change your password right now.
                Please try again later</h1>
            <?php } else { ?>
            <h1>You are currently logged in as "<?php
                    echo $hotspotUsername;
                ?>"</h1>

            <?php if(!empty($errors)) { ?>
            <div id="errors"><ul>
                <?php foreach ($errors as $error) { ?>
                <li><?php echo $error; ?></li>
                <?php } ?>
            </ul></div>
            <?php } elseif (isset($_POST['password'])) { ?>
            <div id="success">Your password has been changed.</div>
            <?php } ?>

            <form action="" method="post">
                <ul>
                    <li>
                        <label for="password">New password:</label>
                        <input type="password" id="password" name="password" value="" />
                    </li>
                    <li>
                        <label for="password2">Confirm new password:</label>
```

```
                        <input type="password" id="password2" name="password2" value="" />
                    </li>
                    <li>
                        <input type="submit" id="act" name="act" value="Change password" />
                    </li>
                </ul>
            </form>
            <?php } ?>
        </div>
    </body>
</html>
```

"Forgotten password" form for hotspot users

The following script needs to be accessible from a server in a walled garden, i.e. users must not need to be logged in to access it. You should link to it from the login page.

To prevent arbitrary people from resetting passwords, the script here requires users to provide two pieces of personal information: Email, and phone. The latter is expected to be the "comment" for a user. If both pieces are correct, the password is set to a new password the user defines.

This scheme is used for the sake of simplicity. Depending on the rest of your setup (e.g. if you have a public trial account, or an SMS gateway...), you may have better ways to deal with confirming the user's identity.

```
<?php
use PEAR2\Net\RouterOS;
require_once 'PEAR2/Autoload.php';

$errors = array();

//Check if the form was submitted. Don't bother with the checks if not.
if (isset($_POST['act'])) {
    try {
        //Adjust RouterOS IP, username and password accordingly.
        $client = new RouterOS\Client('192.168.0.1', 'admin', 'password');
    } catch(Exception $e) {
        $errors[] = $e->getMessage();
    }

    if (empty($_POST['email'])) {
        $errors[] = 'Email is required.';
    }

    if (empty($_POST['phone'])) {
        $errors[] = 'Phone is required.';
    }

    if (empty($errors)) {
        //Check if this is an imposter or not
        $printRequest = new RouterOS\Request('/ip hotspot user print .proplist=.id');
```

```
        $printRequest->setQuery(
            RouterOS\Query::where('email', $_POST['email'])->andWhere('comment', $_POST['phone'])
        );
        $id = $client->sendSync($printRequest)->getArgument('.id');
        if (null === $id) {
            $errors[] = 'Email or phone does not match that of any user.';
        }
    }

    if (!isset($_POST['password']) || !isset($_POST['password2'])) {
        $errors[] = 'Setting a new password is required.';
    }

    if (empty($errors)) {
        if ($_POST['password'] !== $_POST['password2']) {
            $errors[] = 'Passwords do not match.';
        } else {
            //Here's the fun part - actually changing the password
            $setRequest = new RouterOS\Request('/ip hotspot user set');
            $client->sendSync($setRequest
                ->setArgument('password', $_POST['password'])
                ->setArgument('numbers', $id)
            );

            //Redirect back to the login page, thus indicating success.
            header('Location: http://192.168.0.1/login.html');
        }
    }
}

?><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>Forgot your hotspot password?</title>
        <style type="text/css">#errors {background-color:darkred;color:white;}</style>
    </head>
    <body>
        <div>
            <h1>You can reset your hotspot password by filling the following form.
            You'll be redirected back to the login page once you're done</h1>
            <?php if(!empty($errors)) { ?>
            <div id="errors"><ul>
                <?php foreach ($errors as $error) { ?>
                <li><?php echo $error; ?></li>
                <?php } ?>
            </ul></div>
```

```
            <?php } ?>
            <form action="" method="post">
                <ul>
                    <li>
                        <label for="email">Email:</label>
                        <input type="text" id="email" name="email" value="" />
                    </li>
                    <li>
                        <label for="phone">Phone:</label>
                        <input type="text" id="phone" name="phone" value="" />
                    </li>
                    <li>
                        <label for="password">New password:</label>
                        <input type="password" id="password" name="password" value="" />
                    </li>
                    <li>
                        <label for="password2">Confirm new password:</label>
                        <input type="password" id="password2" name="password2" value="" />
                    </li>
                    <li>
                        <input type="submit" id="act" name="act" value="Reset password" />
                    </li>
                </ul>
            </form>
        </div>
    </body>
</html>
```

MAC finder

Tired of asking your customers to tell you their MAC address (and go over the same "click here and..." instructions over and over again)? Well, using the following script, you can now... switch the insructions to "go to this web page... by clicking here and...":

```
<?php
use PEAR2\Net\RouterOS;
require_once 'PEAR2/Autoload.php';

?><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>Your MAC address</title>
    </head>
    <body>
        <h1>
            <?php
try {
```

```
    //Adjust RouterOS IP, username and password accordingly.
    $client = new RouterOS\Client('192.168.0.1', 'admin', 'password');

    $printRequest = new RouterOS\Request('/ip arp print .proplist=mac-address');
    $printRequest->setQuery(
        RouterOS\Query::where('address', $_SERVER['REMOTE_ADDR'])
    );
    $mac = $client->sendSync($printRequest)->getArgument('mac-address');

    if (null !== $mac) {
        echo 'Your MAC address is: ', $mac;
    } else {
        echo 'Your IP (', $_SERVER['REMOTE_ADDR'],
        ") is not part of our network, and because of that, we can't determine your MAC address.";
    }
} catch(Exception $e) {
    echo "We're sorry, but we can't determine your MAC address right now.";
}
?>
        </h1>
    </body>
</html>
```

References

[1] http://pear2.github.com/Net_RouterOS/
[2] http://forum.mikrotik.com/
[3] http://www.gnu.org/copyleft/lesser.html

API in C using winsock

This is an implementation of RouterOS API written on C that using winsock2 for compatibility with Windows operation systems. It's based on API in C [1] (You should use all sources from here, except mikrotik-api.c, that incompatible with Windows). Use compatible mikrotik-api.c source from below.

If your compiler does not support "#pragma comment", add ws2_32.lib to your linker manually.

RouterOS API Source file (mikrotik-api.c)

```
/**************************************************************************

 * Some definitions

 * Word = piece of API code

 * Sentence = multiple words

 * Block = multiple sentences (usually in response to a sentence request)

 *


      int fdSock;

      int iLoginResult;

      struct Sentence stSentence;

      struct Block stBlock;


      fdSock = apiConnect("10.0.0.1", 8728);


      // attempt login

      iLoginResult = login(fdSock, "admin", "adminPassword");


      if (!iLoginResult)

      {

            apiDisconnect(fdSock);

            printf("Invalid username or password.\n");

            exit(1);

      }


      // initialize, fill and send sentence to the API

      initializeSentence(&stSentence);

      addWordToSentence(&stSentence, "/interface/getall");

      writeSentence(fdSock, &stSentence);


      // receive and print block from the API

      stBlock = readBlock(fdSock);

      printBlock(&stBlock);


      apiDisconnect(fdSock);


 **************************************************************************/


// C implementation of Mikrotik's API rewritten for Winsock2 (for windows)
```

```
// Updated 28 August 2011 by hel


#include <stdio.h>

#include <winsock2.h>

#pragma comment(lib, "ws2_32.lib")

#include <windows.h>

#include <string.h>

#include <stdlib.h>

#include "md5.h"

#include "mikrotik-api.h"




/**********************************************************************

 * Connect to API

 * Returns a socket descriptor

 **********************************************************************/

int apiConnect(char *szIPaddr, int iPort)

{


      int fdSock;

      struct sockaddr_in address;

      int iConnectResult;

      int iLen;


      WORD versionWanted = MAKEWORD(1,1);

      WSADATA wsaData;

      WSAStartup(versionWanted, &wsaData);


      fdSock = socket(AF_INET, SOCK_STREAM, 0);


      address.sin_family = AF_INET;

      address.sin_addr.s_addr = inet_addr(szIPaddr);

      address.sin_port = htons(iPort);

      iLen = sizeof(address);


      DEBUG ? printf("Connecting to %s\n", szIPaddr) : 0;


      iConnectResult = connect(fdSock, (struct sockaddr *)&address, iLen);


      if(iConnectResult==-1)

      {

            perror ("Connection problem");

            exit(1);

      }

      else

      {

            DEBUG ? printf("Successfully connected to %s\n", szIPaddr) : 0;
```

```
	}


	// determine endianness of this machine
	// iLittleEndian will be set to 1 if we are
	// on a little endian machine...otherwise
	// we are assumed to be on a big endian processor
	iLittleEndian = isLittleEndian();


	return fdSock;
}




/*********************************************************************
 * Disconnect from API
 * Close the API socket
 *********************************************************************/
void apiDisconnect(int fdSock)
{
	DEBUG ? printf("Closing socket\n") : 0;


	closesocket(fdSock);
}




/*********************************************************************
 * Login to the API
 * 1 is returned on successful login
 * 0 is returned on unsuccessful login
 *********************************************************************/
int login(int fdSock, char *username, char *password)
{
	struct Sentence stReadSentence;
	struct Sentence stWriteSentence;
	char *szMD5Challenge;
	char *szMD5ChallengeBinary;
	char *szMD5PasswordToSend;
	char *szLoginUsernameResponseToSend;
	char *szLoginPasswordResponseToSend;
	md5_state_t state;
	md5_byte_t digest[16];
	char cNull[1] = {0};



	writeWord(fdSock, "/login");
	writeWord(fdSock, "");
```

```
stReadSentence = readSentence(fdSock);
DEBUG ? printSentence (&stReadSentence) : 0;

if (stReadSentence.iReturnValue != DONE)
{
      printf("error.\n");
      exit(0);
}

// extract md5 string from the challenge sentence
szMD5Challenge = strtok(stReadSentence.szSentence[1], "=");
szMD5Challenge = strtok(NULL, "=");

DEBUG ? printf("MD5 of challenge = %s\n", szMD5Challenge) : 0;

// convert szMD5Challenge to binary
szMD5ChallengeBinary = md5ToBinary(szMD5Challenge);

// get md5 of the password + challenge concatenation
md5_init(&state);
md5_append(&state, cNull, 1);
md5_append(&state, (const md5_byte_t *)password, strlen(password));
md5_append(&state, (const md5_byte_t *)szMD5ChallengeBinary, 16);
md5_finish(&state, digest);

// convert this digest to a string representation of the hex values
// digest is the binary format of what we want to send
// szMD5PasswordToSend is the "string" hex format
szMD5PasswordToSend = md5DigestToHexString(digest);

DEBUG ? printf("szPasswordToSend = %s\n", szMD5PasswordToSend) : 0;

// put together the login sentence
initializeSentence(&stWriteSentence);

addWordToSentence(&stWriteSentence, "/login");
addWordToSentence(&stWriteSentence, "=name=");
addPartWordToSentence(&stWriteSentence, username);
addWordToSentence(&stWriteSentence, "=response=00");
addPartWordToSentence(&stWriteSentence, szMD5PasswordToSend);

DEBUG ? printSentence(&stWriteSentence) : 0;
writeSentence(fdSock, &stWriteSentence);



stReadSentence = readSentence(fdSock);
```

```
      DEBUG ? printSentence (&stReadSentence) : 0;


      if (stReadSentence.iReturnValue == DONE)
      {
            return 1;
      }
      else
      {
            return 0;
      }
}




/*****************************************************************
 * Encode message length and write it out to the socket
 *****************************************************************/
void writeLen(int fdSock, int iLen)
{
      char *cEncodedLength;  // encoded length to send to the api socket
      char *cLength;         // exactly what is in memory at &iLen integer


      cLength = calloc(sizeof(int), 1);
      cEncodedLength = calloc(sizeof(int), 1);


      // set cLength address to be same as iLen
      cLength = (char *)&iLen;


      DEBUG ? printf("length of word is %d\n", iLen) : 0;


      // write 1 byte
      if (iLen < 0x80)
      {
            cEncodedLength[0] = (char)iLen;
            send (fdSock, cEncodedLength, 1, 0);
      }


      // write 2 bytes
      else if (iLen < 0x4000)
      {
            DEBUG ? printf("iLen < 0x4000.\n") : 0;


            if (iLittleEndian)
            {
                  cEncodedLength[0] = cLength[1] | 0x80;
                  cEncodedLength[1] = cLength[0];
            }
```

```
        else
        {
              cEncodedLength[0] = cLength[2] | 0x80;
              cEncodedLength[1] = cLength[3];
        }

        send (fdSock, cEncodedLength, 2, 0);
  }

  // write 3 bytes
   else if (iLen < 0x200000)
  {
        DEBUG ? printf("iLen < 0x200000.\n") : 0;

        if (iLittleEndian)
        {
              cEncodedLength[0] = cLength[2] | 0xc0;
              cEncodedLength[1] = cLength[1];
              cEncodedLength[2] = cLength[0];
        }
        else
        {
              cEncodedLength[0] = cLength[1] | 0xc0;
              cEncodedLength[1] = cLength[2];
              cEncodedLength[2] = cLength[3];
        }

        send (fdSock, cEncodedLength, 3, 0);
  }

  // write 4 bytes
  // this code SHOULD work, but is untested...
  else if (iLen < 0x10000000)
  {
        DEBUG ? printf("iLen < 0x10000000.\n") : 0;

        if (iLittleEndian)
        {
              cEncodedLength[0] = cLength[3] | 0xe0;
              cEncodedLength[1] = cLength[2];
              cEncodedLength[2] = cLength[1];
              cEncodedLength[3] = cLength[0];
        }
        else
        {
              cEncodedLength[0] = cLength[0] | 0xe0;
              cEncodedLength[1] = cLength[1];
```

```
                  cEncodedLength[2] = cLength[2];
                  cEncodedLength[3] = cLength[3];
            }

            send (fdSock, cEncodedLength, 4, 0);
      }
      else  // this should never happen
      {
            printf("length of word is %d\n", iLen);
            printf("word is too long.\n");
            exit(1);
      }
}



/*****************************************************************
 * Write a word to the socket
 *****************************************************************/
void writeWord(int fdSock, char *szWord)
{
      DEBUG ? printf("Word to write is %s\n", szWord) : 0;
      writeLen(fdSock, strlen(szWord));
      send(fdSock, szWord, strlen(szWord), 0);
}



/*****************************************************************
 * Write a sentence (multiple words) to the socket
 *****************************************************************/
void writeSentence(int fdSock, struct Sentence *stWriteSentence)
{
      int iIndex;

      if (stWriteSentence->iLength == 0)
      {
            return;
      }

      DEBUG ? printf("Writing sentence\n"): 0;
      DEBUG ? printSentence(stWriteSentence) : 0;

      for (iIndex=0; iIndex<stWriteSentence->iLength; iIndex++)
      {
            writeWord(fdSock, stWriteSentence->szSentence[iIndex]);
      }
```

```
      writeWord(fdSock, "");
}




/*********************************************************************
 * Read a message length from the socket
 *
 * 80 = 10000000 (2 character encoded length)
 * C0 = 11000000 (3 character encoded length)
 * E0 = 11100000 (4 character encoded length)
 *
 * Message length is returned
 ********************************************************************/
int readLen(int fdSock)
{
      char cFirstChar; // first character read from socket
      char *cLength;   // length of next message to read...will be cast to int at the end
      int *iLen;       // calculated length of next message (Cast to int)


      cLength = calloc(sizeof(int), 1);


      DEBUG ? printf("start readLen()\n") : 0;


      recv(fdSock, &cFirstChar, 1, 0);


      DEBUG ? printf("byte1 = %#x\n", cFirstChar) : 0;


      // read 4 bytes
      // this code SHOULD work, but is untested...
      if ((cFirstChar & 0xE0) == 0xE0)
      {
            DEBUG ? printf("4-byte encoded length\n") : 0;


            if (iLittleEndian)
            {
                  cLength[3] = cFirstChar;
                  cLength[3] &= 0x1f;        // mask out the 1st 3 bits
                  recv(fdSock, &cLength[2], 1, 0);
                  recv(fdSock, &cLength[1], 1, 0);
                  recv(fdSock, &cLength[0], 1, 0);
            }
            else
            {
                  cLength[0] = cFirstChar;
                  cLength[0] &= 0x1f;        // mask out the 1st 3 bits
```

```
                recv(fdSock, &cLength[1], 1, 0);
                recv(fdSock, &cLength[2], 1, 0);
                recv(fdSock, &cLength[3], 1, 0);
        }

        iLen = (int *)cLength;
    }

    // read 3 bytes
    else if ((cFirstChar & 0xC0) == 0xC0)
    {
        DEBUG ? printf("3-byte encoded length\n") : 0;

        if (iLittleEndian)
        {
                cLength[2] = cFirstChar;
                cLength[2] &= 0x3f;        // mask out the 1st 2 bits
                recv(fdSock, &cLength[1], 1, 0);
                recv(fdSock, &cLength[0], 1, 0);
        }
        else
        {
                cLength[1] = cFirstChar;
                cLength[1] &= 0x3f;        // mask out the 1st 2 bits
                recv(fdSock, &cLength[2], 1, 0);
                recv(fdSock, &cLength[3], 1, 0);
        }

        iLen = (int *)cLength;
    }

    // read 2 bytes
    else if ((cFirstChar & 0x80) == 0x80)
    {
        DEBUG ? printf("2-byte encoded length\n") : 0;

        if (iLittleEndian)
        {
                cLength[1] = cFirstChar;
                cLength[1] &= 0x7f;        // mask out the 1st bit
                recv(fdSock, &cLength[0], 1, 0);
        }
        else
        {
                cLength[2] = cFirstChar;
                cLength[2] &= 0x7f;        // mask out the 1st bit
                recv(fdSock, &cLength[3], 1, 0);
```

```
            }


            iLen = (int *)cLength;
      }


      // assume 1-byte encoded length...same on both LE and BE systems
      else
      {
            DEBUG ? printf("1-byte encoded length\n") : 0;
            iLen = malloc(sizeof(int));
            *iLen = (int)cFirstChar;
      }


      return *iLen;
}







/*****************************************************************
 * Read a word from the socket
 * The word that was read is returned as a string
 *****************************************************************/
char *readWord(int fdSock)
{
      int iLen = readLen(fdSock);
      int iBytesToRead = 0;
      int iBytesRead = 0;
      char *szWord;
      char *szRetWord;
      char *szTmpWord;


      DEBUG ? printf("readWord iLen=%x\n", iLen) : 0;


      if (iLen > 0)
      {
            // allocate memory for strings
            szRetWord = calloc(sizeof(char), iLen + 1);
            szTmpWord = calloc(sizeof(char), 1024 + 1);


            while (iLen != 0)
            {
                  // determine number of bytes to read this time around
                  // lesser of 1024 or the number of byes left to read
                  // in this word
                  iBytesToRead = iLen > 1024 ? 1024 : iLen;
```

```
                // read iBytesToRead from the socket
                iBytesRead = recv(fdSock, szTmpWord, iBytesToRead, 0);

                // terminate szTmpWord
                szTmpWord[iBytesRead] = 0;

                // concatenate szTmpWord to szRetWord
                strcat(szRetWord, szTmpWord);

                // subtract the number of bytes we just read from iLen
                iLen -= iBytesRead;
          }

          // deallocate szTmpWord
          free(szTmpWord);

          DEBUG ? printf("word = %s\n", szRetWord) : 0;
          return szRetWord;
     }
     else
     {
          return NULL;
     }
}



/*****************************************************************
 * Read a sentence from the socket
 * A Sentence struct is returned
 *****************************************************************/
struct Sentence readSentence(int fdSock)
{
     struct Sentence stReturnSentence;
     char *szWord;
     int i=0;
     int iReturnLength=0;

     DEBUG ? printf("readSentence\n") : 0;

     initializeSentence(&stReturnSentence);

     while (szWord = readWord(fdSock))
     {
          addWordToSentence(&stReturnSentence, szWord);
```

```
            // check to see if we can get a return value from the API
            if (strstr(szWord, "!done") != NULL)
            {
                  DEBUG ? printf("return sentence contains !done\n") : 0;
                  stReturnSentence.iReturnValue = DONE;
            }
            else if (strstr(szWord, "!trap") != NULL)
            {
                  DEBUG ? printf("return sentence contains !trap\n") : 0;
                  stReturnSentence.iReturnValue = TRAP;
            }
            else if (strstr(szWord, "!fatal") != NULL)
            {
                  DEBUG ? printf("return sentence contains !fatal\n") : 0;
                  stReturnSentence.iReturnValue = FATAL;
            }

      }

      // if any errors, get the next sentence
      if (stReturnSentence.iReturnValue == TRAP || stReturnSentence.iReturnValue == FATAL)
      {
            readSentence(fdSock);
      }

      if (DEBUG)
      {
            for (i=0; i<stReturnSentence.iLength; i++)
            {
                  printf("stReturnSentence.szSentence[%d] = %s\n", i, stReturnSentence.szSentence[i]);
            }
      }

      return stReturnSentence;
}



/*****************************************************************
 * Read sentence block from the socket...keeps reading sentences
 * until it encounters !done, !trap or !fatal from the socket
 *****************************************************************/
struct Block readBlock(int fdSock)
{
      struct Sentence stSentence;
    struct Block stBlock;
      initializeBlock(&stBlock);
```

```
	DEBUG ? printf("readBlock\n") : 0;


	do
	{
		stSentence = readSentence(fdSock);
		DEBUG ? printf("readSentence succeeded.\n") : 0;


		addSentenceToBlock(&stBlock, &stSentence);
		DEBUG ? printf("addSentenceToBlock succeeded\n") : 0;


	} while (stSentence.iReturnValue == 0);



	DEBUG ? printf("readBlock completed successfully\n") : 0;


	return stBlock;
}




/*****************************************************************
 * Initialize a new block
 * Set iLength to 0.
 *****************************************************************/
void initializeBlock(struct Block *stBlock)
{
	DEBUG ? printf("initializeBlock\n") : 0;


	stBlock->iLength = 0;
}



/*****************************************************************
 * Clear an existing block
 * Free all sentences in the Block struct and set iLength to 0.
 *****************************************************************/
void clearBlock(struct Block *stBlock)
{
	DEBUG ? printf("clearBlock\n") : 0;


	free(stBlock->stSentence);
	initializeBlock(&stBlock);
}
```

```
/*****************************************************************
 * Print a block.
 * Output a Block with printf.
 *****************************************************************/
void printBlock(struct Block *stBlock)
{
      int i;

      DEBUG ? printf("printBlock\n") : 0;
      DEBUG ? printf("block iLength = %d\n", stBlock->iLength) : 0;

      for (i=0; i<stBlock->iLength; i++)
      {
            printSentence(stBlock->stSentence[i]);
      }
}



/*****************************************************************
 * Add a sentence to a block
 * Allocate memory and add a sentence to a Block.
 *****************************************************************/
void addSentenceToBlock(struct Block *stBlock, struct Sentence *stSentence)
{
      int iNewLength;
      iNewLength = stBlock->iLength + 1;

      DEBUG ? printf("addSentenceToBlock iNewLength=%d\n", iNewLength) : 0;

      // allocate mem for the new Sentence position
      if (stBlock->iLength == 0)
      {
            stBlock->stSentence = malloc(1 * sizeof stBlock->stSentence);
      }
      else
      {
            stBlock->stSentence = realloc(stBlock->stSentence, iNewLength * sizeof stBlock->stSentence + 1);
      }

      // allocate mem for the full sentence struct
      stBlock->stSentence[stBlock->iLength] = malloc(sizeof *stSentence);

      // copy actual sentence struct to the block position
      memcpy(stBlock->stSentence[stBlock->iLength], stSentence, sizeof *stSentence);
```

```
	// update iLength
	stBlock->iLength = iNewLength;

	DEBUG ? printf("addSentenceToBlock stBlock->iLength=%d\n", stBlock->iLength) : 0;
}



/*****************************************************************
 * Initialize a new sentence
 *****************************************************************/
void initializeSentence(struct Sentence *stSentence)
{
	DEBUG ? printf("initializeSentence\n") : 0;

	stSentence->iLength = 0;
	stSentence->iReturnValue = 0;
}



/*****************************************************************
 * Clear an existing sentence
 *****************************************************************/
void clearSentence(struct Sentence *stSentence)
{
	DEBUG ? printf("initializeSentence\n") : 0;

	free(stSentence->szSentence);
	initializeSentence(stSentence);
}



/*****************************************************************
 * Add a word to a sentence struct
 *****************************************************************/
void addWordToSentence(struct Sentence *stSentence, char *szWordToAdd)
{
	int iNewLength;
	iNewLength = stSentence->iLength + 1;

	// allocate mem for the new word position
	if (stSentence->iLength == 0)
	{
		stSentence->szSentence = malloc(1 * sizeof stSentence->szSentence);
	}
```

```
	else
	{
		stSentence->szSentence = realloc(stSentence->szSentence, iNewLength * sizeof stSentence->szSentence + 1);
	}


	// allocate mem for the full word string
	stSentence->szSentence[stSentence->iLength] = malloc(strlen(szWordToAdd) + 1);

	// copy word string to the sentence
	strcpy(stSentence->szSentence[stSentence->iLength], szWordToAdd);

	// update iLength
	stSentence->iLength = iNewLength;
}



/*********************************************************************
 * Add a partial word to a sentence struct...useful for concatenation
 *********************************************************************/
void addPartWordToSentence(struct Sentence *stSentence, char *szWordToAdd)
{
	int iIndex;
	iIndex = stSentence->iLength - 1;

	// reallocate memory for the new partial word
	stSentence->szSentence[iIndex] = realloc(stSentence->szSentence[iIndex], strlen(stSentence->szSentence[iIndex]) + strlen(szWordToAdd) + 1);

	// concatenate the partial word to the existing sentence
	strcat (stSentence->szSentence[iIndex], szWordToAdd);
}



/*********************************************************************
 * Print a Sentence struct
 *********************************************************************/
void printSentence(struct Sentence *stSentence)
{
	int i;

	DEBUG ? printf("Sentence iLength = %d\n", stSentence->iLength) : 0;
	DEBUG ? printf("Sentence iReturnValue = %d\n", stSentence->iReturnValue) : 0;

	printf("Sentence iLength = %d\n", stSentence->iLength);
```

```
	printf("Sentence iReturnValue = %d\n", stSentence->iReturnValue);


	for (i=0; i<stSentence->iLength; i++)
	{
		printf(">>> %s\n", stSentence->szSentence[i]);
	}


	printf("\n");
}




/*****************************************************************
 * MD5 helper function to convert an md5 hex char representation to
 * binary representation.
 *****************************************************************/
char *md5ToBinary(char *szHex)
{
	int di;
	char cBinWork[3];
	char *szReturn;


	// allocate 16 + 1 bytes for our return string
	szReturn = malloc((16 + 1) * sizeof *szReturn);


	// 32 bytes in szHex?
	if (strlen(szHex) != 32)
	{
		return NULL;
	}


	for (di=0; di<32; di+=2)
	{
		cBinWork[0] = szHex[di];
		cBinWork[1] = szHex[di + 1];
		cBinWork[2] = 0;


		DEBUG ? printf("cBinWork = %s\n", cBinWork) : 0;


		szReturn[di/2] = hexStringToChar(cBinWork);
	}


	return szReturn;
}
```

```
/*********************************************************************
 * MD5 helper function to calculate and return hex representation
 * of an MD5 digest stored in binary.
 ********************************************************************/
char *md5DigestToHexString(md5_byte_t *binaryDigest)
{
      int di;
      char *szReturn;

      // allocate 32 + 1 bytes for our return string
      szReturn = malloc((32 + 1) * sizeof *szReturn);



      for (di = 0; di < 16; ++di)
      {
            sprintf(szReturn + di * 2, "%02x", binaryDigest[di]);
      }

      return szReturn;
}




/*********************************************************************
 * Quick and dirty function to convert hex string to char...
 * the toConvert string MUST BE 2 characters + null terminated.
 ********************************************************************/
char hexStringToChar(char *cToConvert)
{
      char cConverted;
      unsigned int iAccumulated=0;
      char cString0[2] = {cToConvert[0], 0};
      char cString1[2] = {cToConvert[1], 0};

      // look @ first char in the 16^1 place
      if (cToConvert[0] == 'f' || cToConvert[0] == 'F')
      {
            iAccumulated += 16*15;
      }
      else if (cToConvert[0] == 'e' || cToConvert[0] == 'E')
      {
            iAccumulated += 16*14;
      }
      else if (cToConvert[0] == 'd' || cToConvert[0] == 'D')
      {
            iAccumulated += 16*13;
      }
```

```
else if (cToConvert[0] == 'c' || cToConvert[0] == 'C')
{
      iAccumulated += 16*12;
}
else if (cToConvert[0] == 'b' || cToConvert[0] == 'B')
{
      iAccumulated += 16*11;
}
else if (cToConvert[0] == 'a' || cToConvert[0] == 'A')
{
      iAccumulated += 16*10;
}
else
{
      iAccumulated += 16 * atoi(cString0);
}



// now look @ the second car in the 16^0 place
if (cToConvert[1] == 'f' || cToConvert[1] == 'F')
{
      iAccumulated += 15;
}
else if (cToConvert[1] == 'e' || cToConvert[1] == 'E')
{
      iAccumulated += 14;
}
else if (cToConvert[1] == 'd' || cToConvert[1] == 'D')
{
      iAccumulated += 13;
}
else if (cToConvert[1] == 'c' || cToConvert[1] == 'C')
{
      iAccumulated += 12;
}
else if (cToConvert[1] == 'b' || cToConvert[1] == 'B')
{
      iAccumulated += 11;
}
else if (cToConvert[1] == 'a' || cToConvert[1] == 'A')
{
      iAccumulated += 10;
}
else
{
      iAccumulated += atoi(cString1);
}
```

```
	DEBUG ? printf("%d\n", iAccumulated) : 0;
	return (char)iAccumulated;
}




/*****************************************************************
 * Test whether or not this system is little endian at RUNTIME
 * Courtesy: http://download.osgeo.org/grass/grass6_progman/endian_8c_source.html
 *****************************************************************/
int isLittleEndian(void)
{
	union
	{
		int testWord;
		char testByte[sizeof(int)];
	} endianTest;

	endianTest.testWord = 1;

	if (endianTest.testByte[0] == 1)
	return 1;		/* true: little endian */

	return 0;		/* false: big endian */
}
```

References

[1] http://wiki.mikrotik.com/wiki/API_in_C

Manual:API Python3

Summary

Since python language have introduced changes to syntax when going from 2.x to 3.x some adjustments had to be made for old code from API.

Code for Python3

code

```
#!/usr/bin/python3

import sys, posix, time, binascii, socket, select
import hashlib

class ApiRos:
    "Routeros api"
    def __init__(self, sk):
        self.sk = sk
        self.currenttag = 0

    def login(self, username, pwd):
        for repl, attrs in self.talk(["/login"]):
            chal = binascii.unhexlify((attrs['=ret']).encode('UTF-8'))
        md = hashlib.md5()
        md.update(b'\x00')
        md.update(pwd.encode('UTF-8'))
        md.update(chal)
        self.talk(["/login", "=name=" + username,
                   "=response=00" + binascii.hexlify(md.digest()).decode('UTF-8') ])

    def talk(self, words):
        if self.writeSentence(words) == 0: return
        r = []
        while 1:
            i = self.readSentence();
            if len(i) == 0: continue
            reply = i[0]
            attrs = {}
            for w in i[1:]:
                j = w.find('=', 1)
                if (j == -1):
                    attrs[w] = ''
                else:
                    attrs[w[:j]] = w[j+1:]
            r.append((reply, attrs))
            if reply == '!done': return r
```

```
def writeSentence(self, words):
    ret = 0
    for w in words:
        self.writeWord(w)
        ret += 1
    self.writeWord('')
    return ret

def readSentence(self):
    r = []
    while 1:
        w = self.readWord()
        if w == '': return r
        r.append(w)

def writeWord(self, w):
    print(("<<< " + w))
    self.writeLen(len(w))
    self.writeStr(w)

def readWord(self):
    ret = self.readStr(self.readLen())
    print((">>> " + ret))
    return ret

def writeLen(self, l):
    if l < 0x80:
        self.writeStr(chr(l))
    elif l < 0x4000:
        l |= 0x8000
        self.writeStr(chr((l >> 8) & 0xFF))
        self.writeStr(chr(l & 0xFF))
    elif l < 0x200000:
        l |= 0xC00000
        self.writeStr(chr((l >> 16) & 0xFF))
        self.writeStr(chr((l >> 8) & 0xFF))
        self.writeStr(chr(l & 0xFF))
    elif l < 0x10000000:
        l |= 0xE0000000
        self.writeStr(chr((l >> 24) & 0xFF))
        self.writeStr(chr((l >> 16) & 0xFF))
        self.writeStr(chr((l >> 8) & 0xFF))
        self.writeStr(chr(l & 0xFF))
    else:
        self.writeStr(chr(0xF0))
        self.writeStr(chr((l >> 24) & 0xFF))
```

```
            self.writeStr(chr((l >> 16) & 0xFF))
            self.writeStr(chr((l >> 8) & 0xFF))
            self.writeStr(chr(l & 0xFF))

    def readLen(self):
        c = ord(self.readStr(1))
        if (c & 0x80) == 0x00:
            pass
        elif (c & 0xC0) == 0x80:
            c &= ~0xC0
            c <<= 8
            c += ord(self.readStr(1))
        elif (c & 0xE0) == 0xC0:
            c &= ~0xE0
            c <<= 8
            c += ord(self.readStr(1))
            c <<= 8
            c += ord(self.readStr(1))
        elif (c & 0xF0) == 0xE0:
            c &= ~0xF0
            c <<= 8
            c += ord(self.readStr(1))
            c <<= 8
            c += ord(self.readStr(1))
            c <<= 8
            c += ord(self.readStr(1))
        elif (c & 0xF8) == 0xF0:
            c = ord(self.readStr(1))
            c <<= 8
            c += ord(self.readStr(1))
            c <<= 8
            c += ord(self.readStr(1))
            c <<= 8
            c += ord(self.readStr(1))
        return c

    def writeStr(self, str):
        n = 0;
        while n < len(str):
            r = self.sk.send(bytes(str[n:], 'UTF-8'))
            if r == 0: raise RuntimeError("connection closed by remote end")
            n += r

    def readStr(self, length):
        ret = ''
        while len(ret) < length:
            s = self.sk.recv(length - len(ret))
```

```
            if s == '': raise RuntimeError("connection closed by remote end")
            ret += s.decode('UTF-8', 'replace')
        return ret

def main():
    s = None
    for res in socket.getaddrinfo(sys.argv[1], "8728", socket.AF_UNSPEC, socket.SOCK_STREAM):
        af, socktype, proto, canonname, sa = res
        try:
             s = socket.socket(af, socktype, proto)
        except (socket.error, msg):
            s = None
            continue
        try:
            s.connect(sa)
        except (socket.error, msg):
            s.close()
            s = None
            continue
        break
    if s is None:
        print ('could not open socket')
        sys.exit(1)

    apiros = ApiRos(s);
    apiros.login(sys.argv[2], sys.argv[3]);

    inputsentence = []

    while 1:
        r = select.select([s, sys.stdin], [], [], None)
        if s in r[0]:
            # something to read in socket, read sentence
            x = apiros.readSentence()

        if sys.stdin in r[0]:
            # read line from input and strip off newline
            l = sys.stdin.readline()
            l = l[:-1]

            # if empty line, send sentence and start with new
            # otherwise append to input sentence
            if l == '':
                apiros.writeSentence(inputsentence)
                inputsentence = []
            else:
                inputsentence.append(l)
```

```
if __name__ == '__main__':
    main()
```

file

lapi client in python3 [1]

References

[1] http://wiki.mikrotik.com/images/6/6b/Api.txt

API multiplatform from townet

MKAPI: LIBRARY TO CONTROL MIKROTIK BOARDS

This article refers to the library developed by townet to control a mikrotik boad using the api, and to the example applications developed using the same library. The library is written in C++, and works under Linux and under Windows.

You can download the lybrary and the SKD, that contains also the manual at the address: www.wispmax.com/media/CPE_SDK.zip

The port used for the mikrotik API is the 8723. On the CPE machine te API needs to be enabled.

1. define API_PORT 8728

This are some functions usable to read the parameters received in a message froma mikrotik board.

AnsiString parse_par(AnsiString stream, char* par, char* dest); AnsiString parse_line(char* reply);

class mikrotikBoard { public:

The one under is the login function, to start the connection. To do the login you needs to give the IP address, the username and the password. You will receive back an empty string it the login is correct, or a string that describe the error that caused the wrong login.

```
 AnsiString login(AnsiString remote, AnsiString user, AnsiString pass);
```

The “command” function is used to handle a transactional command. This function needs to have the command as a parameter, and gives back the reply received from the mikrotik board. If a “trap” is received, this mean that an error occoured, and a messageBox is shown to the user.

```
AnsiString command(AnsiString text);
```

The “xcommand” function allows, like the one before, to execute a transactional command, with the difference that in case of a trap no messagebox giving back the error is generated. The task of handling the error remain to te application’s programmer. The “xcommand” way is more flexible, and useful in case of a command that, may be, was allread given to the board.

```
 AnsiString xcommand(AnsiString text);
```

To handle interactive commands, like for example ‘scan’:

```
void start_icommand(AnsiString cmd);   //Starts an interactive command
 AnsiString icommand();          //Reads part of the reply to the interactive command
 AnsiString end_icommand();    //This is called to close the icommand
```

```
};
```

USE OF MKAPI LIBRARY

To write a C program using MkApi you need to include the library when you compile your application, and yu need to add the source file “MkApi.h” to ypur C sources.

You need to istantiate an object of the class “mikrotikboard”:

```
mikrotikboard mk;
```

Then you need to ask for a login:

```
AnsiString res = mk.login("192.168.1.1", "admin", "secret");
```

If there is an error you can signal it, and for example stop the application:

```
if (res !=0)
{ Application->MessageBox(res.c_str(), "Errore", MB_OK);
  Application->Terminate();
};
```

Now it’s possible to execute commands:

```
AnsiString adrs = mk.command("/ip/address/print");
Application->MessageBox(adrs.c_str(), "Assigned addresses", MB_OK);
```

If you don’t want to receive a popup window in case of error you can use xcommand:

```
AnsiString adrs = mk.xcommand("/ip/address/print");
if (adrs.Pos("!trap") >0) { HANDLE THE ERROR };
Application->MessageBox(adrs.c_str(), "Assigned addresses", MB_OK);
```

You can handle interactive commands in this way:

```
start_icommand("/interface/wireless/scan");
int k;
for (k=0; k<10; k++)
{ AnsiString obj = icommand();
  printf("NEIGHBORS: %s\n", obj.c_str());
};
end_icommand();
```

A routine that execute a command gives back a string containing the reply of the mikrotik board. This string is non modified because the format is not standard, and can be slightly different from a command to another one. So we write two functions: parse_par and parse_line, that can be used to read the parameters returned from a mikrotik board without errors. Here wodn there is an example of code usable to read and write on the computer screen the ip addresses assigned to a mikrotik board.

```
mikrotikboard mk;
AnsiString res=mk.login("192.168.1.1", "admin", "print");
if (res="")
{ AnsiString indirizzi =mk.command("/ip/address/print");
  //This copy the ansistring on a buffer of chars.
  char c[10000]; strcpy(c, indirizzi.c_str());
  while (c[0]!=0)
```

```
  { AnsiString line = parse_line(&c);
    //Here the variable line contains only a line from the reply.
    char adr[30];
    char interf[20];
    parse_par(line, "address=", adr);
    parse_par(line, "interface=", interf);
    printf("Interface %s \taddress %s\n", interf, adr);
  };
};
```

More easily the function parse_par can be used to read the first occourrence of a parameter and to cut the original string, so you can gradually read all the results, in this way:

```
mikrotikboard mk;
AnsiString res=mk.login("192.168.1.1", "admin", "print");
if (res="")
{ AnsiString indirizzi =mk.command("/ip/address/print");

      //This copy the string on a buffer of chars
```

```
  char c[10000]; strcpy(c, indirizzi.c_str());
  while (indirizzi !="")
  { char adr[30];
    char interf[20];
    line = parse_par(line, "address=", adr);
    line=parse_par(line, "interface=", interf);
    printf("Interfaccia %s \tindirizzo %s\n", interf, adr);
  };
};
```

Reading the reply you have to be careful for a thing: if the “interface” information is given in the reply before the “address”, the results can be printed in a wrong position.

The MkApi library define many other functions, but this are aminly low-level ones, and are used to build the ones we descrived before. So we decided not to give documentation about it. If you select to use the “DLL” version of the MkApi library you can write a similar program using any language, like for example basic or c#, without an excessive loose of speed. The same library work also under linux, and can be linked as an object file os as a .so dynamic linked library.

COMMAND LINE UTILITY TO HANDLE MIKROTIK BOARDS

Mik utility is a ms-dos and linux command line application that allows to execute commands on a mikrotik board in a “batch” way. It’s possible to use it from the command line or using scripts written in many languages. Pratically all the programming languages have a shell construct (in basic for example the construct is “system”). SO it’ s possible to execute an external application and then resume the normal execution of a program. So this utility can be used to have a program written in any language that talks with routeros.

Launching the utility mik you will ever have two replies: a first line containing a number followed from the text Err or Ok , and he following lines containing a detailed reply.

If the details of the reply are not importants for the calling application, for example if mik is givin back his internal short manual, this details are not returned on the standard output but on the standard error.

./mik -1 Err Parametri errati

MKDeal - tool di interfaccia mikrotik uso: MKDeal ip utente password -c comando (per comandi one-shot) MKDeal ip utente password -i comando (per gestire comandi interattivi) MKDeal -f nomefile -c comando (per gestire comandi su piu' board)

```
      Il file deve avere come formato IP#USER#PASS su ogni riga
```

risposte: NUM Err oppure NUM Ok seguito dai dati -1 : parametri errati -2 connessione fallita -3 trap generica -4 password errata -5 trap su comando 0 corretto

Some example:

./mik 192.168.1.1 admin rtmtc -c /ip/address/print -2 Err L'indirizzo indicato non risponde

In the upper example the boars is not properly connected to the network or the API is disabled.

$ ./mik 192.168.1.39 admin "" -c /ip/adress/print -5 Err !trap =category=0 =message=no such command or directory (adress)

<STR>!trap =message=no such command prefix

<STR>!done <STR>

In the upper example we have a trap: the command i used is not existing (i wrote address with a single d).

$ ./mik 192.168.1.39 admin "" -c /ip/address/print [*** Errore ***] [This board is not enabled for API Townet]

This is what happens when the board is connected to the network, but the code to enable it for using the API townet is not installed. The MkApi library is in fact working only on townet/wispmax machines. Forother machine it's possible to ask an activation code to townet giving us the serial number and license-id of the board, and the model. There isn't an error code because the API didn't return a result, non allowing the access.

$ ./mik 192.168.1.39 admin "" -c /ip/address/print 0 Ok !re =.id=*3 =comment=aaa =address=10.10.10.1/24 =network=10.10.10.0 =broadcast=10.10.10.255 =interface=ether1

<STR>!re =.id=*4 =comment=bbb =address=192.168.1.39/24 =network=192.168.1.0 =broadcast=192.168.1.255 =interface=ether1

<STR>!done

<STR>

Here is a correct example, and we can see the reply to the command. In this example we have two replies, and there is a tag "<STR>" that is used to show where is the end of a line. Each <STR> close a single reply. The command ever ends with a <STR> describing an empty reply.

$ ./mik 192.168.1.39 admin "" -c /ip/dns/set#=primary-dns=99.99.99.99 0 Ok !done

<STR>

$ ./mik 192.168.1.39 admin "" -c /ip/dns/print 0 Ok !re =primary-dns=99.99.99.99 =secondary-dns=62.94.0.2 =allow-remote-requests=false =max-udp-packet-size=512 =cache-size=2048 =cache-max-ttl=1w00:00:00 =cache-used=5

<STR>!done

<STR>

Now i used a command that has some parameter. For example i'm changing the primary-dns of the board, and then i print the new DNS configuration. The first command is a multirow. To write it on a single row under ms.dos we need to user the # character to signal the end of the line. Under linux, it's a good thing to use the quotation marks around the last parameter.

The "mik" utility is written as a demo for the library MkApi.

An important note: The specification of the API mikrotik defines the command in a format that is slightly different than the ones used from the telnet interface. The ID of the network addresses, and in general of the objects given

back from a command, are ever integer numbers, but this ones doesn't start from the number zero, like working in telnet. To set the address of an ethernet port, for example, you may need to execute a print command to read all the id's, and then to execute the "set" command passing the id of the line you really want to change. All the commands needs to be launched from the root of the command's tree, so "/ip/address/print" is a valid command, while the two commands "/ip/address" and "print" can't be separated. To separater the parts of the commands tou needs to use a slash "/" instead of the space, used in the telnet interface.

To see if there are other differences you can read the documentation of mikrotik api interface at the following address:

[[1]]

DLL MKAPI, TO CONTROL MIKROTIK BOARDS

The mikapi.dll is a dynamic loadable library based on mikrotik api, and allows to connect to a mikrotik board from pratically all the languages under windows. Due to the fact that MkApi is based on the mikrotikboard class and that a DLL is a set of functions and not a set of classes, the DLL hide the MikApi object and shows a list of functions to login, to execute commands and then to logout. The DLL library has the following interface:

AnsiString mk_parse_par(AnsiString stream, char* par, char* dest); AnsiString mk_parse_line(char* reply); AnsiString mk_login(AnsiString remote, AnsiString user, AnsiString pass); AnsiString mk_command(AnsiString text); AnsiString mk_xcommand(AnsiString text); void mk_start_icommand(AnsiString cmd); AnsiString mk_icommand(); AnsiString mk_end_icommand(); void mk_close();

This functions works exactly like the same functions of the original MkApi library. To connect to more than one board it's possible to close the current connection with the instruction mk_close, and then to open a new one. This function is not necessary when the class MkApi is used directly because the mk_close is directly called when the object is deallocated.

References

[1] http://wiki.mikrotik.com/wiki/API

MikroNode

Mikronode

Full-Featured asynchronous Mikrotik API interface for NodeJS [1].

```
var api = require('mikronode');

var connection = new api('192.168.0.1','admin','password');
connection.connect(function(conn) {

   var chan=conn.openChannel();

   chan.write('/ip/address/print',function() {
      chan.on('done',function(data) {

         var parsed = api.parseItems(data);

         parsed.forEach(function(item) {
            console.log('Interface/IP: '+item.interface+"/"+item.address);
         });

         chan.close();
         conn.close();

      });
   });
});
```

Installation

Clone this repository into your node_modules directory. - or -

```
    $ npm install mikronode
```

Features

- Channel based communication
- Multiple channels can be used at once.
- Synchronous execution of commands issued on the same channel.
- Asynchrounous execution of commands issued on different channels.
- Focus on high performance

TODO

- Cleanup login section in connect method.
- Re-design code to hide internal methods and variables.
- Utilize promises.
- Write tests to make sure everything keeps working while making above changes.

API

Connection Object

Calling new api(host,user,pass,options) returns a connection object.

The options argument is optional. If specified, it is an object with these fields:

- **timeout**: number of seconds to wait before timing out due to inactivity.
- **debug**: a value between 0 and 5. Greater value means more verbose.
- **port**: alternative port to connect. (In case it's being mapped with a firewall)
- conn.connect(callback)

  Connect to the target device. The callback function is called after successful login with the current connection object as its parameter.
- conn.openChannel(id)

  Open and return a new channel object. Each channel is a unique command line to the mikrotik, allowing simultaneous execution of commands. The ID parameter is optional.
- conn.isConnected()

  Returns true is currently connected to a mikrotik device.
- conn.closeChannel(id)

  Closes an open channel. This will call the close method of the channel object.
- conn closeOnDone(b)

  If b == true, when a done event occurs, close the connection after all channels have been closed.
- conn.close(force)

  Close the connection. If force is true, force close of any open channels then close this connection.

Channel

The following methods are available for channels:

- channel.closeOnDone(b)

  If b == true, when a done event occurs, close the channel after all commands queued have been executed.
- channel.setSaveBuffer(b)

  If b is true, then save each line received in a buffer and pass the entire buffer to the done event. Otherwise the done event will not get all the lines, only the last line.

  This is handy when following trailing output from a listen command, where the data could be endless.
- channel.getConnection()
- channel.getId()
- channel.write(lines,writeCallback)

  Lines can be a string, or an array of strings. If it is a string, then it is split on the EOL character and each resulting line is sent as a separate word (in API speak) If lines is an array, then each element is sent

unaltered.

- channel.close(force)

  Close the channel. If there are any commands still waiting to be executed, they will be completed before closing the channel.

  If force is TRUE, then the channel is immediately closed. If the channel is running, the cancel command is sent to stop any running listen commands, or potentially long running output.

Examples

Connect to a Mikrotik, and add an address to ether1

```
var api = require('mikronode');

var connection = new api('192.168.0.1','admin','password');
connection.connect(function(conn) {

   var chan=conn.openChannel();

   chan.write(['/ip/address/add','=interface=ether1','=address=192.168.1.1'],function() {
      chan.on('trap',function(data) {
         console.log('Error setting IP: '+data);
      });
      chan.on('done',function(data) {
         console.log('IP Set.');
      });
      chan.close();
      conn.close();
   });
});
```

Writing the program for the example API conversation on the Mikrotik Wiki

```
var api = require('mikronode');

var connection = new api('192.168.0.1','admin','password');
connection.connect(function(conn) {

   conn.closeOnDone(true);
   var chan2=conn.openChannel(2);
   chan2.write('/interface/listen',function(chan) {
      chan.on('read',function(data) {
         packet=api.parseItems([data])[0];
         console.log('Interface change: '+JSON.stringify(packet));
      });
   });

   var chan3=conn.openChannel(3);
   chan3.closeOnDone(true);
```

```
    chan3.write(['/interface/set','=disabled=yes','=.id=ether1'],function(chan) {
       chan.on('done',function(d,chan) {
          // We do this here, 'cause we want channel 4 to write after channel 3 is done.
          var chan4=conn.openChannel(4); // We'll use this later.
          chan4.closeOnDone(true);
          chan4.write(['/interface/set','=disabled=no','=.id=ether1'],function() {
            var chan5=conn.openChannel(5);
            chan5.closeOnDone(true);
            chan5.write('/interface/getall',function(chan) {
               chan.on('done',function(data) {
                  packets=api.parseItems(data);
                  packets.forEach(function(packet) {
                      console.log('Interface: '+JSON.stringify(packet));
                  });
                  chan2.close(); // This should call the /cancel command to stop the listen.
               });
            });
          })
       });
    });
});
```

Simplifying the above by reducing the number of channels.

Notice how the callback embedding is not needed using the syncronous capability.

```
var api = require('mikronode');


var connection = new api('192.168.0.1','admin','password');
connection.connect(function(conn) {


   conn.closeOnDone(true); // All channels need to complete before the connection will close.
   var listenChannel=conn.openChannel();
   listenChannel.write('/interface/listen',function(chan) {
      chan.on('read',function(data) {
         packet=api.parseItems([data])[0];
         console.log('Interface change: '+JSON.stringify(packet));
      });
   });


   var actionChannel=conn.openChannel();
   // These will run synchronsously
   actionChannel.write(['/interface/set','=disabled=yes','=.id=ether1']); // don't care to do anything after it's done.
   actionChannel.write(['/interface/set','=disabled=no','=.id=ether1']); // don't care to do anything after it's done.
   actionChannel.write('/interface/getall',function(chan) {
      chan.on('done',function(data) {
         packets=api.parseItems(data);
         packets.forEach(function(packet) {
```

```
            console.log('Interface: '+JSON.stringify(packet));
        });
        listenChannel.close(); // This should call the /cancel command to stop the listen.
    });
});
actionChannel.close(); // The above commands will complete before this is closed.
});
```

License

(The MIT License)

Copyright (c) 2011 Brandon Myers `trakkasure@gmail.com` [2]

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the 'Software'), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED 'AS IS', WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

External links

- NodeJS [1]

References

[1] http://nodejs.org
[2] mailto:trakkasure@gmail.com

API in Java

Summary

RouterOS API access library written in Java. This code is also capable to connect to IPv6 addresses.

Licensing

Code is provided as is and can be freely used freely. I, as a writer of code, am not responsible for anything that may arise from use of this code.

Usage

Simple example how this can be used. T3apiView class is the interface class that is not supplied here and is mentioned here only do show, how to start simple listener thread to receive the data replied by RouterOS

```
       ApiConnection ret = new ApiConnection("192.168.88.1", 8728);
       if (!ret.isConnected()) {
           ret.start();
           try {
               ret.join();
               if (ret.isConnected()) {
                   ret.login("admin", new char[0]);
               }
           } catch (InterruptedException ex) {
               Logger.getLogger(T3apiView.class.getName()).log(Level.SEVERE, null, ex);
               return null;
           }
       }
       aConn.sendCommand("/ip/address/print");
       DataReceiver dataRec = new DataReceiver(aConn, this);
       dataRec.start();
```

```
import libAPI.*;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author janisk
 */
public class DataReceiver extends Thread {

  private ApiConnection aConn = null;
  T3apiView t3A = null;

  public DataReceiver(ApiConnection aConn, T3apiView t3A) {
      this.aConn = aConn;
      this.t3A = t3A;
```

```
  }

  @Override
  public void run() {
      String s = "";
      while (true) {
          try {
              s = aConn.getData();
              if (s != null) {
                  t3A.outputHere(s);
                  if (s.contains("!done")) {
                  }
              }
          } catch (InterruptedException ex) {
              Logger.getLogger(DataReceiver.class.getName()).log(Level.SEVERE, null, ex);
          }
      }
  }
}
```

Code

Code that is ready to be compiled and used. In some places some comments may be missing. Compiled jar file [1] of same java classes

ApiConn.java

Main file of the package

```
package libAPI;

/*
 * This contains connection. Everything should be here,
 * should operate with this class only
 */

import java.io.*;
import java.net.*;
import java.util.concurrent.LinkedBlockingQueue;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author janisk
 */
public class ApiConn extends Thread {

  private Socket sock = null;
```

```
private DataOutputStream out = null;
private DataInputStream in = null;
private String ipAddress;
private int ipPort;
private boolean connected = false;
private String message = "Not connected";
private ReadCommand readCommand = null;
private WriteCommand writeCommand = null;
private Thread listener = null;
LinkedBlockingQueue queue = new LinkedBlockingQueue(40);

/**
 * Constructor of the connection class
 * @param ipAddress - IP address of the router you want to conenct to
 * @param ipPort - port used for connection, ROS default is 8728
 */
public ApiConn(String ipAddress, int ipPort) {
    this.ipAddress = ipAddress;
    this.ipPort = ipPort;
    this.setName("settings");
}

/**
 * State of connection
 * @return - if connection is established to router it returns true.
 */
public boolean isConnected() {
    return connected;
}
public void disconnect() throws IOException{
    listener.interrupt();
    sock.close();
}
private void listen() {
    if (this.isConnected()) {
        if (readCommand == null) {
            readCommand = new ReadCommand(in, queue);
        }
        listener = new Thread(readCommand);
        listener.setDaemon(true);
        listener.setName("listener");
        listener.start();

    }
}

/**
```

```
     * to get IP address of the connection. Reads data from socket created.
     * @return InetAddress
     */
    public InetAddress getIpAddress() {
        return sock == null ? null : sock.getInetAddress();
    }

    /**
     * returns ip address that socket is asociated with.
     * @return InetAddress
     */
    public InetAddress getLocalIpAddress() {
        return sock == null ? null : sock.getLocalAddress();
    }

    /**
     * Socket remote port number
     * @return
     */
    public int getPort() {
        return sock == null ? null : sock.getPort();
    }

    /**
     * return local prot used by socket
     * @return
     */
    public int getLocalPort() {
        return sock == null ? null : sock.getLocalPort();
    }

    /**
     * Returns status message set up bu class.
     * @return
     */
    public String getMessage() {
        return message;
    }

    /**
     * sets and exectues command (sends it to RouterOS host connected)
     * @param s - command will be sent to RouterOS for example "/ip/address/print\n=follow="
     * @return
     */
    public String sendCommand(String s) {
        return writeCommand.setCommand(s).runCommand();
    }
```

```
/**
 * exeecutes already set command.
 * @return returns status of the command sent
 */
public String runCommand() {
    return writeCommand.runCommand();
}

/**
 * Tries to fech data that is repllied to commands sent. It will wait till it can return something.
 * @return returns data sent by RouterOS
 * @throws java.lang.InterruptedException
 */
public String getData() throws InterruptedException {
    String s = (String) queue.take();
    return s;
}

/**
 * returns command that is set at this moment. And will be exectued if runCommand is exectued.
 * @return
 */
public String getCommand() {
    return writeCommand.getCommand();
}

/**
 * set up method that will log you in
 * @param name - username of the user on the router
 * @param password - password for the user
 * @return
 */
public String login(String name, char[] password) {
    this.sendCommand("/login");
    String s = "a";
    try {
        s = this.getData();
    } catch (InterruptedException ex) {
        Logger.getLogger(ApiConn.class.getName()).log(Level.SEVERE, null, ex);
        return "failed read #1";
    }
    if (!s.contains("!trap") && s.length() > 4) {
        String[] tmp = s.trim().split("\n");
        if (tmp.length > 1) {
            tmp = tmp[1].split("=ret=");
            s = "";
```

```
                String transition = tmp[tmp.length - 1];
                String chal = "";
                chal = Hasher.hexStrToStr("00") + new String(password) + Hasher.hexStrToStr(transition);
                chal = Hasher.hashMD5(chal);
                String m = "/login\n=name=" + name + "\n=response=00" + chal;
                s = this.sendCommand(m);
                try {
                    s = this.getData();
                } catch (InterruptedException ex) {
                    Logger.getLogger(ApiConn.class.getName()).log(Level.SEVERE, null, ex);
                    return "failed read #2";
                }
                if (s.contains("!done")) {
                    if (!s.contains("!trap")) {
                        return "Login successful";
                    }
                }
            }
        return "Login failed";
    }
    @Override
    public void run() {
        try {
            InetAddress ia = InetAddress.getByName(ipAddress);
            if (ia.isReachable(1000)) {
                sock = new Socket(ipAddress, ipPort);
                in = new DataInputStream(sock.getInputStream());
                out = new DataOutputStream(sock.getOutputStream());
                connected = true;
                readCommand = new ReadCommand(in, queue);
                writeCommand = new WriteCommand(out);
                this.listen();
                message = "Connected";
            } else {
                message = "Not reachable";
            }
        } catch (UnknownHostException ex) {
            connected = false;
            message = ex.getMessage();
            ex.printStackTrace();
        } catch (IOException ex) {
            connected = false;
            message = ex.getMessage();
            ex.printStackTrace();
        }
    }
```

```
}
```

Hasher.java

Helper functions to perform some tasks

```
package libAPI;

/*
 * Helper.java
 *
 * Created on 08 June 2007, 11:25
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */

import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

/**
 *
 * @author janisk
 */
public class Hasher {

  /**
   * makes MD5 hash of string for use with RouterOS API
   * @param s - variable to make hacsh from
   * @return
   */
  static public String hashMD5(String s) {
      String md5val = "";
      MessageDigest algorithm = null;
      try {
          algorithm = MessageDigest.getInstance("MD5");
      } catch (NoSuchAlgorithmException nsae) {
          System.out.println("Cannot find digest algorithm");
          System.exit(1);
      }
      byte[] defaultBytes = new byte[s.length()];
      for (int i = 0; i < s.length(); i++) {
          defaultBytes[i] = (byte) (0xFF & s.charAt(i));
      }
      algorithm.reset();
      algorithm.update(defaultBytes);
      byte messageDigest[] = algorithm.digest();
      StringBuffer hexString = new StringBuffer();
      for (int i = 0; i < messageDigest.length; i++) {
```

```
            String hex = Integer.toHexString(0xFF & messageDigest[i]);
            if (hex.length() == 1) {
                hexString.append('0');
            }
            hexString.append(hex);
        }
        return hexString.toString();
    }

    /**
     * converts hex value string to normal strint for use with RouterOS API
     * @param s - hex string to convert to
     * @return - converted string.
     */
    static public String hexStrToStr(String s) {
        String ret = "";
        for (int i = 0; i < s.length(); i += 2) {
            ret += (char) Integer.parseInt(s.substring(i, i + 2), 16);
        }
        return ret;
    }
}
```

ReadCommand.java

This reads returns of the API

```
package libAPI;


/*
 * CommandRead.java
 *
 * Created on 19 June 2007, 10:29
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */


import java.io.*;
import java.util.concurrent.*;
import java.util.logging.Level;
import java.util.logging.Logger;


/**
 *
 * @author janisk
 */
public class ReadCommand implements Runnable {
```

```
private DataInputStream in = null;
LinkedBlockingQueue queue = null;


/**
 * Creates a new instance of CommandRead
 * @param in - Data input stream of socket
 * @param queue - data output inteface
 */
public ReadCommand(DataInputStream in, LinkedBlockingQueue queue) {
    this.in = in;
    this.queue = queue;
}

@Override
public void run() {
    byte b = 0;
    String s = "";
    char ch;
    int a = 0;
    while (true) {
        int sk = 0;
        try {
            a = in.read();
        } catch (IOException ex) {
            return;
        }
        if (a != 0 && a > 0) {
            if (a < 0x80) {
                sk = a;
            } else {
                if (a < 0xC0) {
                    a = a << 8;
                    try {
                        a += in.read();
                    } catch (IOException ex) {
                        return;
                    }
                    sk = a ^ 0x8000;
                } else {
                    if (a < 0xE0) {
                        try {
                            for (int i = 0; i < 2; i++) {
                                a = a << 8;
                                a += in.read();
                            }
                        } catch (IOException ex) {
                            Logger.getLogger(ReadCommand.class.getName()).log(Level.SEVERE, null, ex);
```

```
                            return;
                        }
                        sk = a ^ 0xC00000;
                    } else {
                        if (a < 0xF0) {
                            try {
                                for (int i = 0; i < 3; i++) {
                                    a = a << 8;
                                    a += in.read();
                                }
                            } catch (IOException ex) {
                                Logger.getLogger(ReadCommand.class.getName()).log(Level.SEVERE, null, ex);
                                return;
                            }
                            sk = a ^ 0xE0000000;
                        } else {
                            if (a < 0xF8) {
                                try {
                                    a = 0;
                                    for (int i = 0; i < 5; i++) {
                                        a = a << 8;
                                        a += in.read();
                                    }
                                } catch (IOException ex) {
                                    Logger.getLogger(ReadCommand.class.getName()).log(Level.SEVERE, null, ex);
                                    return;
                                }
                            } else {
                            }
                        }
                    }
                }
            }
            s += "\n";
            byte[] bb = new byte[sk];
            try {
                a = in.read(bb, 0, sk);
            } catch (IOException ex) {
                a = 0;
                ex.printStackTrace();
                return;
            }
            if (a > 0) {
                s += new String(bb);
            }
        } else if (b == -1) {
            System.out.println("Error, it should not happen ever, or connected to wrong port");
```

```
            } else {
                try {
                    queue.put(s);
                } catch (InterruptedException ex) {
                    ex.printStackTrace();
                    System.out.println("exiting reader");
                    return;
                }
                s = "";
            }
        }
    }
}
```

WriteCommand.java

All writing to RouterOS API is done using this.

```
package libAPI;


/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */


import java.io.DataOutputStream;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;


/**
 *
 * @author janisk
 */
public class WriteCommand {


  private byte[] len = {0};
  private DataOutputStream out = null;
  private String command = "";


  WriteCommand(DataOutputStream out, String command) {
      this.out = out;
      this.command = command.replaceAll("\n", "").trim();
  }


  WriteCommand(DataOutputStream out) {
      this.out = out;
  }
```

```
WriteCommand setCommand(String command) {

    this.command = command.trim();

    return this;

}


String getCommand() {

    return command;

}


private byte[] writeLen(String command) {

    Integer i = null;

    String s = "";

    String ret = "";

    if (command.length() < 0x80) {

        i = command.length();

    } else if (command.length() < 0x4000) {

        i = Integer.reverseBytes(command.length() | 0x8000);

    } else if (command.length() < 0x20000) {

        i = Integer.reverseBytes(command.length() | 0xC00000);

    } else if (command.length() < 10000000) {

        i = Integer.reverseBytes(command.length() | 0xE0000000);

    } else {

        i = Integer.reverseBytes(command.length());

    }

    s = Integer.toHexString(i);

    if (s.length() < 2) {

        return new byte[]{i.byteValue()};

    } else {

        for (int j = 0; j < s.length(); j += 2) {

            ret += (char) Integer.parseInt(s.substring(j, j + 2), 16) != 0 ? (char) Integer.parseInt(s.substring(j, j + 2), 16) : "";

        }

    }

    char[] ch = ret.toCharArray();

    return ret.getBytes();

}


String runCommand() {

    try {

        byte[] ret = new byte[0];

        if (!command.contains("\n")) {

            int i = 0;

            byte[] b = writeLen(command);

            int retLen = b.length + command.length() + 1;

            ret = new byte[retLen];

            for (i = 0; i < b.length; i++) {

                ret[i] = b[i];

            }
```

```
                for (byte c : command.getBytes("US-ASCII")) {
                    ret[i++] = c;
                }
            } else {
                String[] str = command.split("\n");
                int i = 1;
                int[] iTmp = new int[str.length];
                for (int a = 0; a < str.length; a++) {
                    iTmp[a] = writeLen(str[a]).length + str[a].length();
                }
                for (int b : iTmp) {
                    i += b;
                }
                ret = new byte[i];
                int counter = 0;
                for (int a = 0; a < str.length; a++) {
                    int j = 0;
                    byte[] b = writeLen(str[a]);
                    for (j = 0; j < b.length; j++) {
                        ret[counter++] = b[j];
                    }
                    for (byte c : str[a].getBytes("US-ASCII")) {
                        ret[counter++] = c;
                    }
                }
            }
            out.write(ret);
            return "Sent successfully";
        } catch (IOException ex) {
            Logger.getLogger(WriteCommand.class.getName()).log(Level.SEVERE, null, ex);
            return "failed";
        }
    }
}
```

References

[1] http://www.mikrotik.com/download/libAPI.jar

API In CPP

This is written in C++. The code is based highly on the code from API In C. I like the way this was done in respect to how easy it is to send a command and get a block of sentences back that are easily parsed.

I have removed all the memory leaks and converted it entirely to C++. There is only a few places using any memory allocation and that is mostly in the encoding as its much easier to do with dynamic char arrays. I have made it so it can be compiled in Xcode for use in Obj C++ and should work fine in any other platform with little or no extra work.

This implementation relies on the MD5 digest calculation functions written by Aladdin Enterprises ([1]). An endian test (big/little endian) is also used courtesy GRASS Development Team ([2]). All functions/libraries used from other sources are available under open licenses such as GNU Public License.

Features: Written using C++ Leak Free Supports *nix Platforms including Mac Sentences will return a map object (so no parsing needed really)

Pre-requisite MD5 calculation function header file (md5.h)

```
/*
  Copyright (C) 1999, 2002 Aladdin Enterprises.  All rights reserved.

  This software is provided 'as-is', without any express or implied
  warranty.  In no event will the authors be held liable for any damages
  arising from the use of this software.

  Permission is granted to anyone to use this software for any purpose,
  including commercial applications, and to alter it and redistribute it
  freely, subject to the following restrictions:

  1. The origin of this software must not be misrepresented; you must not
     claim that you wrote the original software. If you use this software
     in a product, an acknowledgment in the product documentation would be
     appreciated but is not required.
  2. Altered source versions must be plainly marked as such, and must not be
     misrepresented as being the original software.
  3. This notice may not be removed or altered from any source distribution.

  L. Peter Deutsch
  ghost@aladdin.com

 */
/* $Id: md5.h,v 1.4 2002/04/13 19:20:28 lpd Exp $ */
/*
  Independent implementation of MD5 (RFC 1321).

  This code implements the MD5 Algorithm defined in RFC 1321, whose
  text is available at
      http://www.ietf.org/rfc/rfc1321.txt
  The code is derived from the text of the RFC, including the test suite
  (section A.5) but excluding the rest of Appendix A.  It does not include
```

```
  any code or documentation that is identified in the RFC as being
  copyrighted.

  The original and principal author of md5.h is L. Peter Deutsch
  <ghost@aladdin.com>.  Other authors are noted in the change history
  that follows (in reverse chronological order):

  2002-04-13 lpd Removed support for non-ANSI compilers; removed
      references to Ghostscript; clarified derivation from RFC 1321;
      now handles byte order either statically or dynamically.
  1999-11-04 lpd Edited comments slightly for automatic TOC extraction.
  1999-10-18 lpd Fixed typo in header comment (ansi2knr rather than md5);
      added conditionalization for C++ compilation from Martin
      Purschke <purschke@bnl.gov>.
  1999-05-03 lpd Original version.
 */

#ifndef md5_INCLUDED
#  define md5_INCLUDED

/*
 * This package supports both compile-time and run-time determination of CPU
 * byte order.  If ARCH_IS_BIG_ENDIAN is defined as 0, the code will be
 * compiled to run only on little-endian CPUs; if ARCH_IS_BIG_ENDIAN is
 * defined as non-zero, the code will be compiled to run only on big-endian
 * CPUs; if ARCH_IS_BIG_ENDIAN is not defined, the code will be compiled to
 * run on either big- or little-endian CPUs, but will run slightly less
 * efficiently on either one than if ARCH_IS_BIG_ENDIAN is defined.
 */

typedef unsigned char md5_byte_t; /* 8-bit byte */
typedef unsigned int md5_word_t; /* 32-bit word */

/* Define the state of the MD5 Algorithm. */
typedef struct md5_state_s {
    md5_word_t count[2];      /* message length in bits, lsw first */
    md5_word_t abcd[4];            /* digest buffer */
    md5_byte_t buf[64];            /* accumulate block */
} md5_state_t;

#ifdef __cplusplus
extern "C"
{
#endif

/* Initialize the algorithm. */
void md5_init(md5_state_t *pms);
```

```
/* Append a string to the message. */
void md5_append(md5_state_t *pms, const md5_byte_t *data, int nbytes);

/* Finish the message and return the digest. */
void md5_finish(md5_state_t *pms, md5_byte_t digest[16]);

#ifdef __cplusplus
}  /* end extern "C" */
#endif

#endif /* md5_INCLUDED */
```

Pre-requisite MD5 calculation function source file (md5.c)

```
/*
  Copyright (C) 1999, 2000, 2002 Aladdin Enterprises.  All rights reserved.

  This software is provided 'as-is', without any express or implied
  warranty.  In no event will the authors be held liable for any damages
  arising from the use of this software.

  Permission is granted to anyone to use this software for any purpose,
  including commercial applications, and to alter it and redistribute it
  freely, subject to the following restrictions:

  1. The origin of this software must not be misrepresented; you must not
     claim that you wrote the original software. If you use this software
     in a product, an acknowledgment in the product documentation would be
     appreciated but is not required.
  2. Altered source versions must be plainly marked as such, and must not be
     misrepresented as being the original software.
  3. This notice may not be removed or altered from any source distribution.

  L. Peter Deutsch
  ghost@aladdin.com

 */
/* $Id: md5.c,v 1.6 2002/04/13 19:20:28 lpd Exp $ */
/*
  Independent implementation of MD5 (RFC 1321).

  This code implements the MD5 Algorithm defined in RFC 1321, whose
  text is available at
      http://www.ietf.org/rfc/rfc1321.txt
  The code is derived from the text of the RFC, including the test suite
  (section A.5) but excluding the rest of Appendix A.  It does not include
  any code or documentation that is identified in the RFC as being
```

```
  copyrighted.

  The original and principal author of md5.c is L. Peter Deutsch
  <ghost@aladdin.com>.  Other authors are noted in the change history
  that follows (in reverse chronological order):

  2002-04-13 lpd Clarified derivation from RFC 1321; now handles byte order
      either statically or dynamically; added missing #include <string.h>
      in library.
  2002-03-11 lpd Corrected argument list for main(), and added int return
      type, in test program and T value program.
  2002-02-21 lpd Added missing #include <stdio.h> in test program.
  2000-07-03 lpd Patched to eliminate warnings about "constant is
      unsigned in ANSI C, signed in traditional"; made test program
      self-checking.
  1999-11-04 lpd Edited comments slightly for automatic TOC extraction.
  1999-10-18 lpd Fixed typo in header comment (ansi2knr rather than md5).
  1999-05-03 lpd Original version.
 */

#include "md5.h"
#include <string.h>

#undef BYTE_ORDER      /* 1 = big-endian, -1 = little-endian, 0 = unknown */
#ifdef ARCH_IS_BIG_ENDIAN
#  define BYTE_ORDER (ARCH_IS_BIG_ENDIAN ? 1 : -1)
#else
#  define BYTE_ORDER 0
#endif

#define T_MASK ((md5_word_t)~0)
#define T1 /* 0xd76aa478 */ (T_MASK ^ 0x28955b87)
#define T2 /* 0xe8c7b756 */ (T_MASK ^ 0x173848a9)
#define T3    0x242070db
#define T4 /* 0xc1bdceee */ (T_MASK ^ 0x3e423111)
#define T5 /* 0xf57c0faf */ (T_MASK ^ 0x0a83f050)
#define T6    0x4787c62a
#define T7 /* 0xa8304613 */ (T_MASK ^ 0x57cfb9ec)
#define T8 /* 0xfd469501 */ (T_MASK ^ 0x02b96afe)
#define T9    0x698098d8
#define T10 /* 0x8b44f7af */ (T_MASK ^ 0x74bb0850)
#define T11 /* 0xffff5bb1 */ (T_MASK ^ 0x0000a44e)
#define T12 /* 0x895cd7be */ (T_MASK ^ 0x76a32841)
#define T13    0x6b901122
#define T14 /* 0xfd987193 */ (T_MASK ^ 0x02678e6c)
#define T15 /* 0xa679438e */ (T_MASK ^ 0x5986bc71)
#define T16    0x49b40821
```

```
#define T17 /* 0xf61e2562 */ (T_MASK ^ 0x09e1da9d)
#define T18 /* 0xc040b340 */ (T_MASK ^ 0x3fbf4cbf)
#define T19    0x265e5a51
#define T20 /* 0xe9b6c7aa */ (T_MASK ^ 0x16493855)
#define T21 /* 0xd62f105d */ (T_MASK ^ 0x29d0efa2)
#define T22    0x02441453
#define T23 /* 0xd8a1e681 */ (T_MASK ^ 0x275e197e)
#define T24 /* 0xe7d3fbc8 */ (T_MASK ^ 0x182c0437)
#define T25    0x21e1cde6
#define T26 /* 0xc33707d6 */ (T_MASK ^ 0x3cc8f829)
#define T27 /* 0xf4d50d87 */ (T_MASK ^ 0x0b2af278)
#define T28    0x455a14ed
#define T29 /* 0xa9e3e905 */ (T_MASK ^ 0x561c16fa)
#define T30 /* 0xfcefa3f8 */ (T_MASK ^ 0x03105c07)
#define T31    0x676f02d9
#define T32 /* 0x8d2a4c8a */ (T_MASK ^ 0x72d5b375)
#define T33 /* 0xfffa3942 */ (T_MASK ^ 0x0005c6bd)
#define T34 /* 0x8771f681 */ (T_MASK ^ 0x788e097e)
#define T35    0x6d9d6122
#define T36 /* 0xfde5380c */ (T_MASK ^ 0x021ac7f3)
#define T37 /* 0xa4beea44 */ (T_MASK ^ 0x5b4115bb)
#define T38    0x4bdecfa9
#define T39 /* 0xf6bb4b60 */ (T_MASK ^ 0x0944b49f)
#define T40 /* 0xbebfbc70 */ (T_MASK ^ 0x4140438f)
#define T41    0x289b7ec6
#define T42 /* 0xeaa127fa */ (T_MASK ^ 0x155ed805)
#define T43 /* 0xd4ef3085 */ (T_MASK ^ 0x2b10cf7a)
#define T44    0x04881d05
#define T45 /* 0xd9d4d039 */ (T_MASK ^ 0x262b2fc6)
#define T46 /* 0xe6db99e5 */ (T_MASK ^ 0x1924661a)
#define T47    0x1fa27cf8
#define T48 /* 0xc4ac5665 */ (T_MASK ^ 0x3b53a99a)
#define T49 /* 0xf4292244 */ (T_MASK ^ 0x0bd6ddbb)
#define T50    0x432aff97
#define T51 /* 0xab9423a7 */ (T_MASK ^ 0x546bdc58)
#define T52 /* 0xfc93a039 */ (T_MASK ^ 0x036c5fc6)
#define T53    0x655b59c3
#define T54 /* 0x8f0ccc92 */ (T_MASK ^ 0x70f3336d)
#define T55 /* 0xffeff47d */ (T_MASK ^ 0x00100b82)
#define T56 /* 0x85845dd1 */ (T_MASK ^ 0x7a7ba22e)
#define T57    0x6fa87e4f
#define T58 /* 0xfe2ce6e0 */ (T_MASK ^ 0x01d3191f)
#define T59 /* 0xa3014314 */ (T_MASK ^ 0x5cfebceb)
#define T60    0x4e0811a1
#define T61 /* 0xf7537e82 */ (T_MASK ^ 0x08ac817d)
#define T62 /* 0xbd3af235 */ (T_MASK ^ 0x42c50dca)
#define T63    0x2ad7d2bb
```

```
#define T64 /* 0xeb86d391 */ (T_MASK ^ 0x14792c6e)


static void
md5_process(md5_state_t *pms, const md5_byte_t *data /*[64]*/)
{
    md5_word_t
      a = pms->abcd[0], b = pms->abcd[1],
      c = pms->abcd[2], d = pms->abcd[3];
    md5_word_t t;
#if BYTE_ORDER > 0
    /* Define storage only for big-endian CPUs. */
    md5_word_t X[16];
#else
    /* Define storage for little-endian or both types of CPUs. */
    md5_word_t xbuf[16];
    const md5_word_t *X;
#endif

    {
#if BYTE_ORDER == 0
      /*
       * Determine dynamically whether this is a big-endian or
       * little-endian machine, since we can use a more efficient
       * algorithm on the latter.
       */
      static const int w = 1;

      if (*((const md5_byte_t *)&w)) /* dynamic little-endian */
#endif
#if BYTE_ORDER <= 0           /* little-endian */
      {
          /*
           * On little-endian machines, we can process properly aligned
           * data without copying it.
           */
          if (!((data - (const md5_byte_t *)0) & 3)) {
            /* data are properly aligned */
            X = (const md5_word_t *)data;
          } else {
            /* not aligned */
            memcpy(xbuf, data, 64);
            X = xbuf;
          }
      }
#endif
#if BYTE_ORDER == 0
```

```
      else                  /* dynamic big-endian */
#endif
#if BYTE_ORDER >= 0           /* big-endian */
      {
          /*
           * On big-endian machines, we must arrange the bytes in the
           * right order.
           */
          const md5_byte_t *xp = data;
          int i;

#  if BYTE_ORDER == 0
          X = xbuf;             /* (dynamic only) */
#  else
#    define xbuf X            /* (static only) */
#  endif
          for (i = 0; i < 16; ++i, xp += 4)
            xbuf[i] = xp[0] + (xp[1] << 8) + (xp[2] << 16) + (xp[3] << 24);
      }
#endif
    }

#define ROTATE_LEFT(x, n) (((x) << (n)) | ((x) >> (32 - (n))))

    /* Round 1. */
    /* Let [abcd k s i] denote the operation
       a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s). */
#define F(x, y, z) (((x) & (y)) | (~(x) & (z)))
#define SET(a, b, c, d, k, s, Ti)\
  t = a + F(b,c,d) + X[k] + Ti;\
  a = ROTATE_LEFT(t, s) + b
    /* Do the following 16 operations. */
    SET(a, b, c, d,  0,  7,  T1);
    SET(d, a, b, c,  1, 12,  T2);
    SET(c, d, a, b,  2, 17,  T3);
    SET(b, c, d, a,  3, 22,  T4);
    SET(a, b, c, d,  4,  7,  T5);
    SET(d, a, b, c,  5, 12,  T6);
    SET(c, d, a, b,  6, 17,  T7);
    SET(b, c, d, a,  7, 22,  T8);
    SET(a, b, c, d,  8,  7,  T9);
    SET(d, a, b, c,  9, 12, T10);
    SET(c, d, a, b, 10, 17, T11);
    SET(b, c, d, a, 11, 22, T12);
    SET(a, b, c, d, 12,  7, T13);
    SET(d, a, b, c, 13, 12, T14);
    SET(c, d, a, b, 14, 17, T15);
```

```
    SET(b, c, d, a, 15, 22, T16);
#undef SET

     /* Round 2. */
     /* Let [abcd k s i] denote the operation
          a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s). */
#define G(x, y, z) (((x) & (z)) | ((y) & ~(z)))
#define SET(a, b, c, d, k, s, Ti)\
  t = a + G(b,c,d) + X[k] + Ti;\
  a = ROTATE_LEFT(t, s) + b
     /* Do the following 16 operations. */
    SET(a, b, c, d,  1,  5, T17);
    SET(d, a, b, c,  6,  9, T18);
    SET(c, d, a, b, 11, 14, T19);
    SET(b, c, d, a,  0, 20, T20);
    SET(a, b, c, d,  5,  5, T21);
    SET(d, a, b, c, 10,  9, T22);
    SET(c, d, a, b, 15, 14, T23);
    SET(b, c, d, a,  4, 20, T24);
    SET(a, b, c, d,  9,  5, T25);
    SET(d, a, b, c, 14,  9, T26);
    SET(c, d, a, b,  3, 14, T27);
    SET(b, c, d, a,  8, 20, T28);
    SET(a, b, c, d, 13,  5, T29);
    SET(d, a, b, c,  2,  9, T30);
    SET(c, d, a, b,  7, 14, T31);
    SET(b, c, d, a, 12, 20, T32);
#undef SET

     /* Round 3. */
     /* Let [abcd k s t] denote the operation
          a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s). */
#define H(x, y, z) ((x) ^ (y) ^ (z))
#define SET(a, b, c, d, k, s, Ti)\
  t = a + H(b,c,d) + X[k] + Ti;\
  a = ROTATE_LEFT(t, s) + b
     /* Do the following 16 operations. */
    SET(a, b, c, d,  5,  4, T33);
    SET(d, a, b, c,  8, 11, T34);
    SET(c, d, a, b, 11, 16, T35);
    SET(b, c, d, a, 14, 23, T36);
    SET(a, b, c, d,  1,  4, T37);
    SET(d, a, b, c,  4, 11, T38);
    SET(c, d, a, b,  7, 16, T39);
    SET(b, c, d, a, 10, 23, T40);
    SET(a, b, c, d, 13,  4, T41);
    SET(d, a, b, c,  0, 11, T42);
```

```
    SET(c, d, a, b,  3, 16, T43);
    SET(b, c, d, a,  6, 23, T44);
    SET(a, b, c, d,  9,  4, T45);
    SET(d, a, b, c, 12, 11, T46);
    SET(c, d, a, b, 15, 16, T47);
    SET(b, c, d, a,  2, 23, T48);
#undef SET

     /* Round 4. */
     /* Let [abcd k s t] denote the operation
          a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s). */
#define I(x, y, z) ((y) ^ ((x) | ~(z)))
#define SET(a, b, c, d, k, s, Ti)\
  t = a + I(b,c,d) + X[k] + Ti;\
  a = ROTATE_LEFT(t, s) + b
     /* Do the following 16 operations. */
    SET(a, b, c, d,  0,  6, T49);
    SET(d, a, b, c,  7, 10, T50);
    SET(c, d, a, b, 14, 15, T51);
    SET(b, c, d, a,  5, 21, T52);
    SET(a, b, c, d, 12,  6, T53);
    SET(d, a, b, c,  3, 10, T54);
    SET(c, d, a, b, 10, 15, T55);
    SET(b, c, d, a,  1, 21, T56);
    SET(a, b, c, d,  8,  6, T57);
    SET(d, a, b, c, 15, 10, T58);
    SET(c, d, a, b,  6, 15, T59);
    SET(b, c, d, a, 13, 21, T60);
    SET(a, b, c, d,  4,  6, T61);
    SET(d, a, b, c, 11, 10, T62);
    SET(c, d, a, b,  2, 15, T63);
    SET(b, c, d, a,  9, 21, T64);
#undef SET

     /* Then perform the following additions. (That is increment each
        of the four registers by the value it had before this block
        was started.) */
    pms->abcd[0] += a;
    pms->abcd[1] += b;
    pms->abcd[2] += c;
    pms->abcd[3] += d;
}

void
md5_init(md5_state_t *pms)
{
    pms->count[0] = pms->count[1] = 0;
```

```
    pms->abcd[0] = 0x67452301;
    pms->abcd[1] = /*0xefcdab89*/ T_MASK ^ 0x10325476;
    pms->abcd[2] = /*0x98badcfe*/ T_MASK ^ 0x67452301;
    pms->abcd[3] = 0x10325476;
}

void
md5_append(md5_state_t *pms, const md5_byte_t *data, int nbytes)
{
    const md5_byte_t *p = data;
    int left = nbytes;
    int offset = (pms->count[0] >> 3) & 63;
    md5_word_t nbits = (md5_word_t)(nbytes << 3);

    if (nbytes <= 0)
      return;

    /* Update the message length. */
    pms->count[1] += nbytes >> 29;
    pms->count[0] += nbits;
    if (pms->count[0] < nbits)
      pms->count[1]++;

    /* Process an initial partial block. */
    if (offset) {
      int copy = (offset + nbytes > 64 ? 64 - offset : nbytes);

      memcpy(pms->buf + offset, p, copy);
      if (offset + copy < 64)
          return;
      p += copy;
      left -= copy;
      md5_process(pms, pms->buf);
    }

    /* Process full blocks. */
    for (; left >= 64; p += 64, left -= 64)
      md5_process(pms, p);

    /* Process a final partial block. */
    if (left)
      memcpy(pms->buf, p, left);
}

void
md5_finish(md5_state_t *pms, md5_byte_t digest[16])
{
```

```
    static const md5_byte_t pad[64] = {
      0x80, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    };
    md5_byte_t data[8];
    int i;

    /* Save the length before padding. */
    for (i = 0; i < 8; ++i)
      data[i] = (md5_byte_t)(pms->count[i >> 2] >> ((i & 3) << 3));
    /* Pad to 56 bytes mod 64. */
    md5_append(pms, pad, ((55 - (pms->count[0] >> 3)) & 63) + 1);
    /* Append the length. */
    md5_append(pms, data, 8);
    for (i = 0; i < 16; ++i)
      digest[i] = (md5_byte_t)(pms->abcd[i >> 2] >> ((i & 3) << 3));
}
```

Mikrotik API Types header file (MikrotikAPITypes.h)

This is the API Types header file. This file contains the Sentence and Block classes.

Notes:

- DEBUG flag is defined for debugging purposes...generates alot of internal data via printf
- NONE, DONE, TRAP and FATAL constants are defined
- Use void GetMap(int index, std::map<std::string, std::string> &sentenceMap); method for getting a map of words for a sentence.
  - Each word in a sentence is stored as a string. Sentence structs contain individual API words (stored as an vector of strings).
  - Block structs represent the full API response...an array of sentences. Blocks are not defined in the Mikrotik API specs, but are a convenient way to represent a full API response in the context of this implementation.

```
//
//  MikrotikAPITypes.h
//  WinboxMobile
//
//  Created by Joey Gentry on 2/11/10.
//  Copyright 2010 __MyCompanyName__. All rights reserved.
//

#include <vector>
#include <map>
#include <string>

#define DEBUG 1

#define NONE 0
```

```
#define DONE 1
#define TRAP 2
#define FATAL 3

class Sentence {
      std::vector<std::string> strWords;    // vecor of strings representing individual words
      int returnType;     // return type of sentence
      void Tokenize(const std::string &str, std::vector<std::string> &tokens,
                            const std::string &delimiters = " ");


public:
      void SetReturnType(int returnTypeIn) { returnType = returnTypeIn; }
      int GetReturnType() { return returnType; }
      void AddWord(const std::string &strWordToAdd) { strWords.push_back(strWordToAdd); }
      void Clear() { strWords.clear(); returnType = 0; }
      int Length() { return strWords.size(); }
      std::string operator[](int index) { return strWords[index]; }
      std::string GetWord(int index) { return strWords[index]; }
      void GetMap(int index, std::map<std::string, std::string> &sentenceMap);
      bool Print();
};


class Block {
      std::vector<Sentence> sentences;

public:
      int Length() { return sentences.size(); }
      void AddSentence(const Sentence &sentence);
      void Clear() { sentences.clear(); }
      Sentence operator[](int index) { return sentences[index]; }
      bool Print();
};
```

Mikrotik API Types source file (MikrotikAPITypes.cpp)

```
//
//  untitled.mm
//  WinboxMobile
//
//  Created by Joey Gentry on 2/13/10.
//  Copyright 2010 __MyCompanyName__. All rights reserved.
//

#include "MikrotikAPITypes.h"

using namespace std;
```

```
/**********************************************************************
 * Print a sentence.
 **********************************************************************/
bool Sentence::Print()
{
      DEBUG ? printf("Sentence Word Count = %d\n", strWords.size()) : 0;
      DEBUG ? printf("Sentence returnType = %d\n", returnType) : 0;

      for (int i = 0; i < strWords.size(); ++i) {
            printf("%s\n", strWords[i].c_str());
      }

      printf("\n");

      return true;
}

void Sentence::GetMap(map<string, string> &sentenceMap)
{
      for (int i = 0; i < strWords.size(); ++i) {
            string tmpDataString = strWords[i];
            vector<string> dataStrings;
            Tokenize(tmpDataString, dataStrings, "=");

            if (returnType == NONE && dataStrings.size() > 1) {
                  sentenceMap.insert(make_pair(dataStrings[1], dataStrings[2]));
            }
      }
}

void Sentence::Tokenize(const string &str, vector<string> &tokens, const string &delimiters)
{
      // Skip delimiters at beginning.
    string::size_type lastPos = str.find_first_not_of(delimiters, 0);
    // Find first "non-delimiter".
    string::size_type pos     = str.find_first_of(delimiters, lastPos);

    while (string::npos != pos || string::npos != lastPos)
    {
        // Found a token, add it to the vector.
        tokens.push_back(str.substr(lastPos, pos - lastPos));
        // Skip delimiters.  Note the "not_of"
        lastPos = str.find_first_not_of(delimiters, pos);
        // Find next "non-delimiter"
        pos = str.find_first_of(delimiters, lastPos);
    }
```

```
}

/*********************************************************************
 * Add a Sentence to a block
 ********************************************************************/
void Block::AddSentence(const Sentence &sentence)
{
      sentences.push_back(sentence);

      DEBUG ? printf("AddSentenceToBlock Size=%d\n", sentences.size()) : 0;
}

/*********************************************************************
 * Print a block.
 ********************************************************************/
bool Block::Print()
{
      DEBUG ? printf("PrintBlock\n") : 0;
      DEBUG ? printf("Block Size = %d\n", sentences.size()) : 0;

      for (int i = 0; i < sentences.size(); ++i) {
            sentences[i].Print();
      }

      return true;
}
```

Mikrotik API header file (MikrotikAPI.h)

This is the API header file. This file contains the MikrotikAPI class.

Notes:

- NOCONNECT, NOLOGIN constants are defined.

```
/*********************************************************************
 * Some definitions
 * Word = piece of API code
 * Sentence = multiple words
 * Block = multiple sentences (usually in response to a Sentence request)
 *

 try {
      MikrotikAPI mt = MikrotikAPI("64.126.135.214", "test", "joey", 8728);

      Sentence sentence;
      Block block;

      // Fill and send sentence to the API
      sentence.AddWord("/interface/getall");
```

```
	mt.WriteSentence(sentence);

	// receive and print block from the API
	mt.ReadBlock(block);
	block.Print();
} catch (int e) {
	if(e == NOCONNECT)
		printf("Could not connect.\n");
	if(e == NOLOGIN)
		printf("Could not login.\n");
}


Original Author: Joey Gentry (www.Murderdev.com)
Feel freel to ask me questions but I can't guarantee I will be able to answer them all.
No warranties are provided with this
code. This was written/converted for my iPhone app to allow accessing a Mikrotik router.
The app is called Winbox Mobile for those who are interested.

This is written in C++. The code is based highly on the code from
 http://wiki.mikrotik.com/wiki/API_in_C. I like the way this was done in respect to how easy
 it is to send a command and get a block of sentences back that are easily parsed.

I have removed all the memory leaks and converted it entirely to C++. There is only a few
places using any memory allocation and that is mostly in the encoding as its much easier
to do with dynamic char arrays. I have made it so it can be compiled in Xcode for use in
Obj C++ and should work fine in any other platform with little or no extra work.

This implementation relies on the MD5 digest calculation functions written by
Aladdin Enterprises http://sourceforge.net/projects/libmd5-rfc/files/. An endian test
(big/little endian) is also used courtesy GRASS Development Team
http://download.osgeo.org/grass/grass6_progman/endian_8c.html. All functions/libraries used
from other sources are available under open licenses such as GNU Public License.

Features:
Written using C++
Leak Free
Supports *nix Platforms including Mac
Sentences will return a map object (so no parsing needed really)

*****************************************************************/

#include<sys/socket.h>
#include<arpa/inet.h>
#include <vector>
#include <string>
#include "md5.h"
```

```
#include "MikrotikAPITypes.h"

#define NOCONNECT 1
#define NOLOGIN 2

class MikrotikAPI  {
      private:
            int fdSock;
            bool littleEndian;

            bool IsLittleEndian();

            void Connect(const std::string &strIpAddress, int port);
            void Disconnect();

            // Word
            void WriteLength(int messageLength);
            int ReadLength();
            void WriteWord(const std::string &strWord);
            void ReadWord(std::string &strWordOut);

            // MD5 helper functions
            std::string MD5DigestToHexString(md5_byte_t *binaryDigest);
            std::string MD5ToBinary(const std::string &strHex);
            char HexStringToChar(const std::string &hexToConvert);

      public:
      MikrotikAPI();
            MikrotikAPI(const std::string &strIpAddress, const std::string &strUsername,
                              const std::string &strPassword, int port);
            ~MikrotikAPI();

            // API specific functions
            int Login(const std::string &strUsername, const std::string &strPassword);

            // Sentence
            void WriteSentence(Sentence &writeSentence);
            void ReadSentence(Sentence &sentenceOut);

            // Block
            void ReadBlock(Block &block);
};
```

Mikrotik API source file (MikrotikAPI.cpp)

```
#include "MikrotikAPI.h"

using namespace std;

MikrotikAPI::MikrotikAPI(const string &strIpAddress, const string &strUsername,
                                    const string &strPassword, int port)
{
      Connect(strIpAddress, port);

      if(fdSock != -1) {

            // attempt login
            int loginResult = Login(strUsername, strPassword);

            if (!loginResult) {
                  throw NOLOGIN;
                  Disconnect();
                  printf("Invalid username or password.\n");
            } else {
                  printf("Logged in successfully.\n");
            }
      } else {
            throw NOCONNECT;
      }
}

MikrotikAPI::~MikrotikAPI()
{
      if(fdSock != -1)
            Disconnect();
}

/**********************************************************************
 * Connect to API
 * Returns a socket descriptor
 **********************************************************************/
void MikrotikAPI::Connect(const string &strIpAddress, int port)
{
      struct sockaddr_in address;
      int connectResult;
      int addressSize;

      fdSock = socket(AF_INET, SOCK_STREAM, 0);

      address.sin_family = AF_INET;
      address.sin_addr.s_addr = inet_addr(strIpAddress.c_str());
```

```
	address.sin_port = htons(port);
	addressSize = sizeof(address);

	DEBUG ? printf("Connecting to %s\n", strIpAddress.c_str()) : 0;

	connectResult = connect(fdSock, (struct sockaddr *)&address, addressSize);

	if(connectResult==-1) {
		perror ("Connection problem");
		Disconnect();
		fdSock = -1;
	} else {
		DEBUG ? printf("Successfully connected to %s\n", strIpAddress.c_str()) : 0;
	}

	// determine endianness of this machine
	// iLittleEndian will be set to 1 if we are
	// on a little endian machine...otherwise
	// we are assumed to be on a big endian processor
	littleEndian = IsLittleEndian();
}

/**************************************************************************
 * Disconnect from API
 * Close the API socket
 *************************************************************************/
void MikrotikAPI::Disconnect()
{
	if(fdSock != -1) {
		DEBUG ? printf("Closing socket\n") : 0;

		close(fdSock);
	}
}

/**************************************************************************
 * Login to the API
 * 1 is returned on successful login
 * 0 is returned on unsuccessful login
 *************************************************************************/
int MikrotikAPI::Login(const string &strUsername, const string &strPassword)
{
	Sentence readSentence;
	Sentence writeSentence;

	md5_state_t state;
	md5_byte_t digest[16];
```

```
char cNull[1] = {0};

//Send login message
WriteWord("/login");
WriteWord(cNull);

ReadSentence(readSentence);
DEBUG ? readSentence.Print() : 0;

if (readSentence.GetReturnType() != DONE) {
      printf("Error.\n");
} else {

      // extract md5 string from the challenge sentence
      char *strWord = new char [readSentence[1].size() + 1];
      strcpy(strWord, readSentence[1].c_str());
      char *md5Challenge = strtok(strWord, "=");
      md5Challenge = strtok(NULL, "=");

      DEBUG ? printf("MD5 of challenge = %s\n", md5Challenge) : 0;

      ////Place of interest: Check to see if this md5Challenge string works as as string.
          //   It may not because it needs to be binary.
      // convert szMD5Challenge to binary
      string md5ChallengeBinary = MD5ToBinary(md5Challenge);
      delete[] strWord;

      // get md5 of the password + challenge concatenation
      md5_init(&state);
      md5_append(&state, (const md5_byte_t *)cNull, 1);
      md5_append(&state, (const md5_byte_t *)strPassword.c_str(),
                          strlen(strPassword.c_str()));
      md5_append(&state, (const md5_byte_t *)md5ChallengeBinary.c_str(), 16);
      md5_finish(&state, digest);

      // convert this digest to a string representation of the hex values
      // digest is the binary format of what we want to send
      // szMD5PasswordToSend is the "string" hex format
      string md5PasswordToSend = MD5DigestToHexString(digest);

      DEBUG ? printf("MD5 Password To Send = %s\n", md5PasswordToSend.c_str()) : 0;

      // put together the login sentence
      writeSentence.AddWord("/login");
      writeSentence.AddWord("=name=" + strUsername);
      writeSentence.AddWord("=response=00" + md5PasswordToSend);
```

```
            DEBUG ? writeSentence.Print() : 0;
            WriteSentence(writeSentence);

            ReadSentence(readSentence);
            DEBUG ? readSentence.Print() : 0;

            if (readSentence.GetReturnType() == DONE) {
                  return 1;
            }
      }
      return 0;
}

/**************************************************************************
 * Encode message length and write it out to the socket
 **************************************************************************/
void MikrotikAPI::WriteLength(int messageLength)
{
      char *encodedLengthData;  // encoded length to send to the api socket
      char *lengthData;     // exactly what is in memory at &iLen integer

      encodedLengthData = (char *)calloc(sizeof(int), 1);

      // set cLength address to be same as messageLength
      lengthData = (char *)&messageLength;

      DEBUG ? printf("Length of word is %d\n", messageLength) : 0;

      // write 1 byte
      if (messageLength < 0x80) {
            encodedLengthData[0] = (char)messageLength;
            write (fdSock, encodedLengthData, 1);
      } else if (messageLength < 0x4000) { // write 2 bytes
            DEBUG ? printf("messageLength < 0x4000.\n") : 0;

            if (littleEndian) {
                  encodedLengthData[0] = lengthData[1] | 0x80;
                  encodedLengthData[1] = lengthData[0];
            } else {
                  encodedLengthData[0] = lengthData[2] | 0x80;
                  encodedLengthData[1] = lengthData[3];
            }

            write (fdSock, encodedLengthData, 2);
      } else if (messageLength < 0x200000) { // write 3 bytes
            DEBUG ? printf("messageLength < 0x200000.\n") : 0;
```

```
            if (littleEndian) {
                  encodedLengthData[0] = lengthData[2] | 0xc0;
                  encodedLengthData[1] = lengthData[1];
                  encodedLengthData[2] = lengthData[0];
            } else {
                  encodedLengthData[0] = lengthData[1] | 0xc0;
                  encodedLengthData[1] = lengthData[2];
                  encodedLengthData[2] = lengthData[3];
            }

            write (fdSock, encodedLengthData, 3);
      } else if (messageLength < 0x10000000) { // write 4 bytes (untested)
            DEBUG ? printf("messageLength < 0x10000000.\n") : 0;

            if (littleEndian) {
                  encodedLengthData[0] = lengthData[3] | 0xe0;
                  encodedLengthData[1] = lengthData[2];
                  encodedLengthData[2] = lengthData[1];
                  encodedLengthData[3] = lengthData[0];
            } else {
                  encodedLengthData[0] = lengthData[0] | 0xe0;
                  encodedLengthData[1] = lengthData[1];
                  encodedLengthData[2] = lengthData[2];
                  encodedLengthData[3] = lengthData[3];
            }

            write (fdSock, encodedLengthData, 4);
      } else  { // this should never happen
            printf("Length of word is %d\n", messageLength);
            printf("Word is too long.\n");
      }

      delete [] encodedLengthData;
}

/******************************************************************
 * Write a word to the socket
 ******************************************************************/
void MikrotikAPI::WriteWord(const string &strWord)
{
      DEBUG ? printf("Word to write is %s\n", strWord.c_str()) : 0;
      WriteLength(strWord.length());
      write(fdSock, strWord.c_str(), strWord.length());
}

/******************************************************************
 * Write a Sentence (multiple words) to the socket
```

```
 ********************************************************************/
void MikrotikAPI::WriteSentence(Sentence &writeSentence)
{
	if (writeSentence.Length() == 0) {
		return;
	}

	DEBUG ? printf("Writing sentence\n"): 0;
	DEBUG ? writeSentence.Print() : 0;

	for (int i = 0; i < writeSentence.Length(); ++i) {
		WriteWord(writeSentence[i]);
	}

	WriteWord("\0");
}

/**********************************************************************
 * Read a message length from the socket
 *
 * 80 = 10000000 (2 character encoded length)
 * C0 = 11000000 (3 character encoded length)
 * E0 = 11100000 (4 character encoded length)
 *
 * Message length is returned
 **********************************************************************/
int MikrotikAPI::ReadLength()
{
	char firstChar; // first character read from socket
	char *lengthData;   // length of next message to read...will be cast to int at the end
	int *messageLength;       // calculated length of next message (Cast to int)

	lengthData = (char *) calloc(sizeof(int), 1);

	DEBUG ? printf("Start ReadLength()\n") : 0;

	read(fdSock, &firstChar, 1);

	DEBUG ? printf("byte1 = %#x\n", firstChar) : 0;

	// read 4 bytes
	// this code SHOULD work, but is untested...
	if ((firstChar & 0xE0) == 0xE0) {
		DEBUG ? printf("4-byte encoded length\n") : 0;

		if (littleEndian){
			lengthData[3] = firstChar;
```

```
                lengthData[3] &= 0x1f;        // mask out the 1st 3 bits
                read(fdSock, &lengthData[2], 1);
                read(fdSock, &lengthData[1], 1);
                read(fdSock, &lengthData[0], 1);
        } else {
                lengthData[0] = firstChar;
                lengthData[0] &= 0x1f;        // mask out the 1st 3 bits
                read(fdSock, &lengthData[1], 1);
                read(fdSock, &lengthData[2], 1);
                read(fdSock, &lengthData[3], 1);
        }

        messageLength = (int *)lengthData;
    } else if ((firstChar & 0xC0) == 0xC0) { // read 3 bytes
        DEBUG ? printf("3-byte encoded length\n") : 0;

        if (littleEndian) {
                lengthData[2] = firstChar;
                lengthData[2] &= 0x3f;        // mask out the 1st 2 bits
                read(fdSock, &lengthData[1], 1);
                read(fdSock, &lengthData[0], 1);
        } else {
                lengthData[1] = firstChar;
                lengthData[1] &= 0x3f;        // mask out the 1st 2 bits
                read(fdSock, &lengthData[2], 1);
                read(fdSock, &lengthData[3], 1);
        }

        messageLength = (int *)lengthData;
    } else if ((firstChar & 0x80) == 0x80) { // read 2 bytes
        DEBUG ? printf("2-byte encoded length\n") : 0;

        if (littleEndian) {
                lengthData[1] = firstChar;
                lengthData[1] &= 0x7f;        // mask out the 1st bit
                read(fdSock, &lengthData[0], 1);
        } else {
                lengthData[2] = firstChar;
                lengthData[2] &= 0x7f;        // mask out the 1st bit
                read(fdSock, &lengthData[3], 1);
        }

        messageLength = (int *)lengthData;
    } else { // assume 1-byte encoded length...same on both LE and BE systems
        DEBUG ? printf("1-byte encoded length\n") : 0;
        messageLength = (int *) malloc(sizeof(int));
        *messageLength = (int)firstChar;
```

```
	}

	int retMessageLength = *messageLength;
	delete messageLength;
	delete [] lengthData;

	return retMessageLength;
}

/*****************************************************************
 * Read a word from the socket
 * The word that was read is returned as a string
 *****************************************************************/
void MikrotikAPI::ReadWord(string &strWordOut)
{
	int messageLength = ReadLength();
	int bytesToRead = 0;
	int bytesRead = 0;

	char *tmpWord;

	DEBUG ? printf("ReadWord messageLength=%x\n", messageLength) : 0;

	strWordOut.clear();
	if (messageLength > 0) {
		// allocate memory for strings
		tmpWord = (char *) calloc(sizeof(char), 1024 + 1);

		while (messageLength != 0) {
			// determine number of bytes to read this time around
			// lesser of 1024 or the number of byes left to read
			// in this word
			bytesToRead = messageLength > 1024 ? 1024 : messageLength;

			// read iBytesToRead from the socket
			bytesRead = read(fdSock, tmpWord, bytesToRead);

			// terminate szTmpWord
			tmpWord[bytesRead] = 0;

			// concatenate szTmpWord to szRetWord
			strWordOut += tmpWord;

			// subtract the number of bytes we just read from iLen
			messageLength -= bytesRead;
		}
```

```
            // deallocate szTmpWord
            delete [] tmpWord;

            DEBUG ? printf("Word = %s\n", strWordOut.c_str()) : 0;
      }
}

/*****************************************************************
 * Read a Sentence from the socket
 * A Sentence struct is returned
 *****************************************************************/
void MikrotikAPI::ReadSentence(Sentence &sentenceOut)
{
      DEBUG ? printf("ReadSentence\n") : 0;

      sentenceOut.Clear();

      string strWord;
      ReadWord(strWord);
      while (!strWord.empty()) {
            sentenceOut.AddWord(strWord);

            // check to see if we can get a return value from the API
            if (strWord.find("!done") != string::npos) {
                  DEBUG ? printf("return Sentence contains !done\n") : 0;
                  sentenceOut.SetReturnType(DONE);
            } else if (strWord.find("!trap") != string::npos) {
                  DEBUG ? printf("return Sentence contains !trap\n") : 0;
                  sentenceOut.SetReturnType(TRAP);
            } else if (strWord.find("!fatal") != string::npos) {
                  DEBUG ? printf("return Sentence contains !fatal\n") : 0;
                  sentenceOut.SetReturnType(FATAL);
            }

            ReadWord(strWord);
      }

      // if any errors, get the next sentence
      if (sentenceOut.GetReturnType() == TRAP || sentenceOut.GetReturnType() == FATAL) {
            ReadSentence(sentenceOut);
      }

      if (DEBUG) {
            for (int i = 0; i < sentenceOut.Length(); ++i) {
                  printf("stReturnSentence.szSentence[%d] = %s\n", i, sentenceOut[i].c_str());
            }
      }
```

```
}

/*****************************************************************
 * Read Sentence Block from the socket...keeps reading sentences
 * until it encounters !done, !trap or !fatal from the socket
 *****************************************************************/
void MikrotikAPI::ReadBlock(Block &block)
{
      Sentence sentence;
    block.Clear();

      DEBUG ? printf("ReadBlock\n") : 0;

      do {
            ReadSentence(sentence);
            DEBUG ? printf("ReadSentence succeeded.\n") : 0;

            block.AddSentence(sentence);
            DEBUG ? printf("AddSentenceToBlock succeeded\n") : 0;
      } while (sentence.GetReturnType() == NONE);

      DEBUG ? printf("ReadBlock completed successfully\n") : 0;
}



/*****************************************************************
 * MD5 helper function to convert an md5 hex char representation to
 * binary representation.
 *****************************************************************/
string MikrotikAPI::MD5ToBinary(const string &strHex)
{
      string strReturn;

      // 32 bytes in szHex?
      if (strHex.length() != 32) {
            return strReturn;
      }

      char binWork[3];
      for (int i = 0; i < 32; i += 2) {
            binWork[0] = strHex[i];
            binWork[1] = strHex[i + 1];
            binWork[2] = 0;

            DEBUG ? printf("binWork = %s\n", binWork) : 0;

            strReturn[i / 2] = HexStringToChar(binWork);
```

```
	}

	return strReturn;
}

/*****************************************************************
 * MD5 helper function to calculate and return hex representation
 * of an MD5 digest stored in binary.
 *****************************************************************/
string MikrotikAPI::MD5DigestToHexString(md5_byte_t *binaryDigest)
{
	char strReturn[32 + 1];

	for (int i = 0; i < 16; ++i) {
		sprintf(strReturn + i * 2, "%02x", binaryDigest[i]);
	}

	return strReturn;
}

/*****************************************************************
 * Quick and dirty function to convert hex string to char...
 * the toConvert string MUST BE 2 characters + null terminated.
 *****************************************************************/
char MikrotikAPI::HexStringToChar(const string &hexToConvert)
{
	unsigned int accumulated = 0;
	char char0[2] = {hexToConvert[0], 0};
	char char1[2] = {hexToConvert[1], 0};

	// look @ first char in the 16^1 place
	if (hexToConvert[0] == 'f' || hexToConvert[0] == 'F') {
		accumulated += 16*15;
	} else if (hexToConvert[0] == 'e' || hexToConvert[0] == 'E') {
		accumulated += 16*14;
	} else if (hexToConvert[0] == 'd' || hexToConvert[0] == 'D') {
		accumulated += 16*13;
	} else if (hexToConvert[0] == 'c' || hexToConvert[0] == 'C') {
		accumulated += 16*12;
	} else if (hexToConvert[0] == 'b' || hexToConvert[0] == 'B') {
		accumulated += 16*11;
	} else if (hexToConvert[0] == 'a' || hexToConvert[0] == 'A') {
		accumulated += 16*10;
	} else {
		accumulated += 16 * atoi(char0);
	}
```

```
	// now look @ the second car in the 16^0 place
	if (hexToConvert[1] == 'f' || hexToConvert[1] == 'F') {
		accumulated += 15;
	} else if (hexToConvert[1] == 'e' || hexToConvert[1] == 'E') {
		accumulated += 14;
	} else if (hexToConvert[1] == 'd' || hexToConvert[1] == 'D') {
		accumulated += 13;
	} else if (hexToConvert[1] == 'c' || hexToConvert[1] == 'C') {
		accumulated += 12;
	} else if (hexToConvert[1] == 'b' || hexToConvert[1] == 'B') {
		accumulated += 11;
	} else if (hexToConvert[1] == 'a' || hexToConvert[1] == 'A') {
		accumulated += 10;
	} else {
		accumulated += atoi(char1);
	}

	DEBUG ? printf("%d\n", accumulated) : 0;
	return (char)accumulated;
}

/*****************************************************************
 * Test whether or not this system is little endian at RUNTIME
 * Courtesy: http://download.osgeo.org/grass/grass6_progman/endian_8c_source.html
 *****************************************************************/
bool MikrotikAPI::IsLittleEndian()
{
	union {
		int testWord;
		char testByte[sizeof(int)];
	} endianTest;

	endianTest.testWord = 1;

	if (endianTest.testByte[0] == 1)
		return 1;		/* true: little endian */

	return 0;		/* false: big endian */
}
```

References

[1] http://sourceforge.net/projects/libmd5-rfc/files/

[2] http://download.osgeo.org/grass/grass6_progman/endian_8c.html

Api php template

This is a php template for working with RouterOS v3 API.

Requirements

1. It uses the php api Class found here : link API_PHP_class [1].

2. It is presented first in the forum here link Forum_link [2]

3. overLIB popup JavaScript library is required for this and its included in the files (zip file) or can be found here:[3]

Connected Clients (from the registration table)

```
<?php

require('routeros_api.class.php');

$API = new routeros_api();

$API->debug = false;

if ($API->connect('192.168.1.2', 'api', 'api1234')) {      // Change this as necessery


 $API->write('/interface/wireless/registration-table/print',false);
 $API->write('=count-only=');

   $READ = $API->read(false);
   $ARRAY = $API->parse_response($READ);


      echo "Number of connected clients:" . substr($READ[1],5);

   $API->disconnect();

}

?>
```

Resources (cpu/mem/disk/version)

```
<?php

require('routeros_api.class.php');

$API = new routeros_api();

$API->debug = false;
```

```
if ($API->connect('192.168.1.2', 'api', 'api1234')) {   // Change
this as necessery

$ARRAY = $API->comm("/system/resource/print");


   $first = $ARRAY['0'];
$memperc = ($first['free-memory']/$first['total-memory']);
$hddperc = ($first['free-hdd-space']/$first['total-hdd-space']);
$mem = ($memperc*100);
$hdd = ($hddperc*100);
echo "Mikrotik RouterOs 4.16 Resources";
echo "<br />";
echo "<table width=550 border=0 align=center>";

echo "<tr><td>Platform, board name and Ros version
is:</td><td>" . $first['platform'] . " - " .
$first['board-name'] . " - "  . $first['version'] . " - " .
$first['architecture-name'] . "</td></tr><br />";
echo "<tr><td>Cpu and available
cores:</td><td>" . $first['cpu'] . " at " .
$first['cpu-frequency'] . " Mhz with " . $first['cpu-count'] . "
core(s) "  . "</td></tr><br />";
echo "<tr><td>Uptime is:</td><td>" .
$first['uptime'] . " (hh/mm/ss)" . "</td></tr><br
/>";
echo "<tr><td>Cpu Load is:</td><td>" .
$first['cpu-load'] . " %" . "</td></tr><br />";
echo "<tr><td>Total,free memory and memory %
is:</td><td>" . $first['total-memory'] . "Kb - " .
$first['free-memory'] . "Kb - " . number_format($mem,3) . "%
</td></tr><br />";
echo "<tr><td>Total,free disk and disk %
is:</td><td>" . $first['total-hdd-space'] . "Kb - " .
$first['free-hdd-space'] . "Kb - " . number_format($hdd,3) . "%
</td></tr><br />";
echo "<tr><td>Sectors (write,since reboot,bad
blocks):</td><td>" . $first['write-sect-total'] . " - " .
$first['write-sect-since-reboot'] . " - " . $first['bad-blocks'] . "%
</td></tr><br />";

echo "</table>";

echo "<br />";
echo "<br />";
echo "<br />";
echo "<br />Debug:";
```

```
echo "<br />";

   $API->disconnect();

}

?>
```

Registration Tables

```
<?php
  function popup( $text, $popup )
  {
  ?>
  <a href="javascript:void(0);" onmouseover="return
overlib('<?php echo($popup); ?>
  ');" onmouseout="return nd();"><?php echo($text);
?></a>
  <?php
  }
  ?>

  <script type="text/javascript" src="overlib/overlib.js"><!--
 overLIB (c) Erik Bosrup -->
  </script>

<?php

require('routeros_api.class.php');

$API = new routeros_api();

$API->debug = false;

if ($API->connect('192.168.1.2', 'api', 'api1234')) {    // Change
this as necessery

$ARRAY = $API->comm("/interface/wireless/registration-table/print");



echo "<table width=100% border=1>";

echo "<tr><td align=left size=2>Id</td><td
size=2>iface</td><td size=2>mac-address</td><td
 size=1>Ap</td><td
size=1>wds</td><td>rx-rate</td><td>tx-rate</td><td>Data</td><td>uptime</td><td>Last
 Activity</td><td>signal
strength</td><td>signal to
```

```
noise</td><td>strength at rates</td><td>tx
ccq</td><td>pthroughput</td><td>ack
timeout</td><td>last ip</td><td>802.1x port
en.</td><td>authentication
type</td><td>encryption</td><td>group
encryption</td><td>wmm</td></tr>";

echo "<tr><td align=left>";


for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" . $regtable['.id'] .
"</font><br>";
            }

echo "</td><td>";


for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" . $regtable['interface']
 . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" .
$regtable['mac-address'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
```

```
$regtable = $ARRAY[$i];
      if ($regtable['ap']=="true")
         {
         echo "<font color=#04B404 size=2>" . $regtable['ap'] .
"</font><br>";
         }else{
         echo "<font color=#FF0000 size=2>". $regtable['ap']
."</font><br>";
         }

            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
            if ($regtable['wds']=="true")
         {
         echo "<font color=#04B404 size=2>" . $regtable['wds'] .
"</font><br>";
         }else{
         echo "<font color=#FF0000 size=2>". $regtable['wds']
."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#000099 size=2>" . $regtable['rx-rate'] .
 "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" . $regtable['tx-rate'] .
```

```
 "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
      echo popup('Data', 'Packets  ' . $regtable['packets'] .
'<br/>Bytes  ' . $regtable['bytes'] . '<br/>Frames  ' .
$regtable['frames'] . '<br/>Frame-Bytes  ' .
$regtable['frame-bytes'] . '<br/>hw-frames  ' .
$regtable['hw-frames'] . '<br/>hw-frame-bytes  ' .
$regtable['hw-frame-bytes'] . '<br/>tx-frames-timed-out  ' .
$regtable['tx-frames-timed-out']);

            }

echo "</td><td>";


for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#003300 size=2>" . $regtable['uptime'] .
"</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#003300 size=2>" .
$regtable['last-activity'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
```

```
$regtable = $ARRAY[$i];

      echo "<font color=#880000 size=2>" .
$regtable['signal-strength'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#A00000 size=2>" .
$regtable['signal-to-noise'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
$z=$regtable['wds'];

if ($z==true)
      {
    echo    popup('Rates', $regtable['strength-at-rates'] ) ;
      }
      if ($z==false)
      {
    echo    popup('Rates', $regtable['strength-at-rates'] ) ;
      }
      else
      {
      echo " ";
      }
      }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" . $regtable['tx-ccq'] .
```

```
"</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" .
$regtable['p-throughput'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" .
$regtable['ack-timeout'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" . $regtable['last-ip'] .
 "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
                        if ($regtable['802.1x-port-enabled']=="true")
         {
         echo "<font color=#04B404 size=2>" .
$regtable['802.1x-port-enabled'] . "</font><br>";
         }else{
```

```
         echo "<font color=#FF0000 size=2>".
$regtable['802.1x-port-enabled'] ."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#CC0000 size=2>" .
$regtable['authentication-type'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#CC0000 size=2>" .
$regtable['encryption'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#CC0000 size=2>" .
$regtable['group-encryption'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
                  if ($regtable['wmm-enabled']=="true")
         {
         echo "<font color=#04B404 size=2>" .
```

```
$regtable['wmm-enabled'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=2>".
$regtable['wmm-enabled'] ."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#CC0000 size=2>" . $regtable['comment'] .
 "</font><br>";
            }

echo "</td><td>";
echo "</table>";
echo "<br />Debug:";
echo "<br />";


   $API->disconnect();

}

?>
```

Basic Interface List

```
<?php
  function popup( $text, $popup )
  {
  ?>
  <a href="javascript:void(0);" onmouseover="return
overlib('<?php echo($popup); ?>
  ');" onmouseout="return nd();"><?php echo($text);
?></a>
  <?php
  }
  ?>

  <script type="text/javascript" src="overlib/overlib.js"><!--
 overLIB (c) Erik Bosrup -->
  </script>
```

```
<?php

require('routeros_api.class.php');

$API = new routeros_api();

$API->debug = false;

if ($API->connect('192.168.1.2', 'api', 'api1234')) {

$ARRAY = $API->comm("/interface/getall");



echo "<table width=100% border=1>";

echo "<tr><td align=left size=3>Id</td><td
size=3>name</td><td size=3>type</td><td
size=3>dynamic</td><td
size=3>disabled</td><td>mtu</td><td>l2mtu</td><td>comment</td></tr>";

echo "<tr><td align=left>";


for ($i=0; $i<20; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=3>" . $regtable['.id'] .
"</font><br>";
            }

echo "</td><td>";


for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=3>" . $regtable['name'] .
"</font><br>";
            }

echo "</td><td>";
```

```
for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=3>" . $regtable['type'] .
"</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
      if ($regtable['dynamic']=="true")
         {
         echo "<font color=#04B404 size=3>" .
$regtable['dynamic'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=3>". $regtable['dynamic']
 ."</font><br>";
         }

            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
            if ($regtable['disabled']=="true")
         {
         echo "<font color=#04B404 size=3>" .
$regtable['disabled'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=3>".
$regtable['disabled'] ."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
```

```
$regtable = $ARRAY[$i];

      echo "<font color=#000099 size=3>" . $regtable['mtu'] .
"</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=3>" . $regtable['l2mtu'] .
"</font><br>";
            }

echo "</td><td>";


echo "</table>";
echo "<br />Debug:";
echo "<br />";
print_r($ARRAY);
   $API->disconnect();

}

?>
```

Wireless Interface List

```
<?php
  function popup( $text, $popup )
  {
  ?>
  <a href="javascript:void(0);" onmouseover="return
overlib('<?php echo($popup); ?>
  ');" onmouseout="return nd();"><?php echo($text);
?></a>
  <?php
  }
  ?>

  <script type="text/javascript" src="overlib/overlib.js"><!--
 overLIB (c) Erik Bosrup -->
  </script>
```

```
<?php

require('routeros_api.class.php');

$API = new routeros_api();

$API->debug = false;

if ($API->connect('192.168.1.2', 'api', 'api1234')) {

$ARRAY = $API->comm("/interface/wireless/print");

echo "<table width=100% border=1>";

echo "<tr><td align=left size=2>Id</td><td
size=2>name</td><td size=2>mtu</td><td
size=1>mac-address</td><td size=1>arp</td><td
width=100 size=5>interface
type</td><td>mode</td><td>ssid</td><td>frequency</td><td>band</td><td>scan
 list</td><td>antenna mode</td><td>wds
mode</td><td>wds default bridge</td><td>wds
ignore ssid</td><td>default
authentication</td><td>default
forwarding</td><td>default ap tx
limit</td><td>default client tx
limit</td><td>hide ssid</td><td>security
profile</td><td>compression</td></tr>";

echo "<tr><td align=left>";


for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" . $regtable['.id'] .
"</font><br>";
            }

echo "</td><td>";


for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
```

```
      echo "<font color=#04B404 size=2>" . $regtable['name'] .
"</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" . $regtable['mtu'] .
"</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

         echo "<font color=#04B404 size=2>" .
$regtable['mac-address'] . "</font><br>";

            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
                  if ($regtable['arp']=="true")
         {
         echo "<font color=#04B404 size=2>" . $regtable['arp'] .
"</font><br>";
         }else{
         echo "<font color=#FF0000 size=2>". $regtable['arp']
."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)
```

```
      {
$regtable = $ARRAY[$i];

      echo "<font color=#000099 size=1>" .
$regtable['interface-type'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" . $regtable['mode'] .
"</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#003300 size=2>" . $regtable['ssid'] .
"</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#003300 size=2>" . $regtable['frequency']
 . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
```

```
      echo "<font color=#880000 size=2>" . $regtable['band'] .
"</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#A00000 size=2>" . $regtable['scan-list']
 . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=2>" .
$regtable['antenna-mode'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
      echo popup('WDS', 'WDS Mode  ' . $regtable['wds-mode'] .
'<br/>WDS Default Bridge  ' . $regtable['wds-default-bridge'] .
'<br/>WDS Ignore SSID  ' . $regtable['wds-ignore-ssid']);

            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
                  if ($regtable['default-authentication']=="true")
         {
```

```
         echo "<font color=#04B404 size=2>" .
$regtable['default-authentication'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=2>".
$regtable['default-authentication'] ."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
                  if ($regtable['default-forwarding']=="true")
         {
         echo "<font color=#04B404 size=2>" .
$regtable['default-forwarding'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=2>".
$regtable['default-forwarding'] ."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

        echo "<font color=#FF0000 size=2>".
$regtable['default-ap-tx-limit'] ."</font><br>";

            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

        echo "<font color=#FF0000 size=2>".
$regtable['default-client-tx-limit'] ."</font><br>";

            }
```

```
echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
                        if ($regtable['hide-ssid']=="true")
         {
         echo "<font color=#04B404 size=2>" .
$regtable['hide-ssid'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=2>".
$regtable['hide-ssid'] ."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#CC0000 size=2>" .
$regtable['security-profile'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
                        if ($regtable['compression']=="true")
         {
         echo "<font color=#04B404 size=2>" .
$regtable['compression'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=2>".
$regtable['compression'] ."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)
```

```
      {
$regtable = $ARRAY[$i];
                          if ($regtable['running']=="true")
         {
         echo "<font color=#04B404 size=2>" .
$regtable['running'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=2>". $regtable['running']
 ."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
                  if ($regtable['disabled']=="true")
         {
         echo "<font color=#04B404 size=2>" .
$regtable['disabled'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=2>".
$regtable['disabled'] ."</font><br>";
         }
            }

echo "</td><td>";

echo "</table>";
echo "<br />Debug:";
echo "<br />";
print_r($ARRAY);

   $API->disconnect();

}

?>
```

Hotspot Hosts List

```
<?php
  function popup( $text, $popup )
  {
  ?>
  <a href="javascript:void(0);" onmouseover="return
overlib('<?php echo($popup); ?>
  ');" onmouseout="return nd();"><?php echo($text);
?></a>
  <?php
  }
  ?>

  <script type="text/javascript" src="overlib/overlib.js"><!--
 overLIB (c) Erik Bosrup -->
  </script>

<?php

require('routeros_api.class.php');

$API = new routeros_api();

$API->debug = false;

if ($API->connect('192.168.1.2', 'api', 'api1234')) {

$ARRAY = $API->comm("/ip/hotspot/host/print");


echo "<table width=100% border=1>";

echo "<tr><td align=left size=3>Id</td><td
size=3>mac-address</td><td
size=3>address</td><td
size=3>to-address</td><td>server</td><td>uptime</td><td>keepalive-timeout</td><td>found-b

echo "<tr><td align=left>";


for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=3>" . $regtable['.id'] .
"</font><br>";
```

```
            }

echo "</td><td>";


for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=3>" .
$regtable['mac-address'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=3>" . $regtable['address'] .
 "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

         echo "<font color=#04B404 size=3>" .
$regtable['to-address'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#000099 size=3>" . $regtable['server'] .
"</font><br>";
            }
```

```
echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#003300 size=3>" . $regtable['uptime'] .
"</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#003300 size=3>" .
$regtable['keepalive-timeout'] . "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#880000 size=3>" . $regtable['found-by']
. "</font><br>";
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
            if ($regtable['DHCP']=="true")
         {
         echo "<font color=#04B404 size=3>" . $regtable['DHCP'] .
 "</font><br>";
         }else{
         echo "<font color=#FF0000 size=3>". $regtable['DHCP']
."</font><br>";
         }
```

```
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
            if ($regtable['authorized']=="true")
         {
         echo "<font color=#04B404 size=3>" .
$regtable['authorized'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=3>".
$regtable['authorized'] ."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];
            if ($regtable['bypassed']=="true")
         {
         echo "<font color=#04B404 size=3>" .
$regtable['bypassed'] . "</font><br>";
         }else{
         echo "<font color=#FF0000 size=3>".
$regtable['bypassed'] ."</font><br>";
         }
            }

echo "</td><td>";

for ($i=0; $i<250; $i++)

      {
$regtable = $ARRAY[$i];

      echo "<font color=#04B404 size=3>" . $regtable['comment'] .
 "</font><br>";
            }

echo "</td><td>";
```

```
echo "</table>";
echo "<br />Debug:";
echo "<br />";
print_r($ARRAY);

   $API->disconnect();

}

?>
```

References

[1] http://wiki.mikrotik.com/wiki/API_PHP_class
[2] http://forum.mikrotik.com/viewtopic.php?f=9&t=50176
[3] http://www.bosrup.com/web/overlib/?Download

API in VB dot NET

This is VB.NET class for connecting and working with Mikrotik API. It will give you basic connectivity so you can log in and send and receive commands.

Class

```
Public Class Mikrotik

      Dim tcpStream As IO.Stream

      Dim tcpCon As New Net.Sockets.TcpClient


      Public Sub New(ByVal ipOrDns As String, Optional ByVal port As Integer = -1)

            Dim ips = Net.Dns.GetHostEntry(ipOrDns)


            tcpCon.Connect(ips.AddressList(0), If(port = -1, 8728, port))

            tcpStream = tcpCon.GetStream()

      End Sub


      Public Sub New(ByVal endP As System.Net.IPEndPoint)

            tcpCon.Connect(endP)

            tcpStream = tcpCon.GetStream()

      End Sub


      Public Sub Close()

            tcpStream.Close()

            tcpCon.Close()

      End Sub


      Public Function Login(ByVal user As String, ByVal pass As String) As Boolean

            Send("/login", True)

            Dim hash = Read()(0).Split(New String() {"ret="}, StringSplitOptions.None)(1)
```

```
        Send("/login")
        Send("=name=" + user)
        Send("=response=00" + EncodePassword(pass, hash), True)
        Dim res = Read()
        If (res(0) = "!done") Then Return True Else Return False
    End Function

    Function EncodePassword(ByVal pass As String, ByVal challange As String) As String
        Dim hash_byte(challange.Length / 2 - 1) As Byte
        For i = 0 To challange.Length - 2 Step 2
            hash_byte(i / 2) = Byte.Parse(challange.Substring(i, 2), Globalization.NumberStyles.HexNumber)
        Next
        Dim response(pass.Length + hash_byte.Length) As Byte
        response(0) = 0
        Text.Encoding.ASCII.GetBytes(pass.ToCharArray()).CopyTo(response, 1)
        hash_byte.CopyTo(response, 1 + pass.Length)



        Dim md5 = New System.Security.Cryptography.MD5CryptoServiceProvider()


        Dim hash = md5.ComputeHash(response)


        Dim hashStr As New Text.StringBuilder()
        For Each h In hash
            hashStr.Append(h.ToString("x2"))
        Next
        Return hashStr.ToString()
    End Function


    Public Sub Send(ByVal command As String, Optional ByVal EndSentence As Boolean = False)
        Dim bytes = System.Text.Encoding.ASCII.GetBytes(command.ToCharArray())
        Dim size = EncodeLength(bytes.Length)


        tcpStream.Write(size, 0, size.Length)
        tcpStream.Write(bytes, 0, bytes.Length)
        If EndSentence Then tcpStream.WriteByte(0)
    End Sub


    Public Function Read() As List(Of String)
        Dim output As New List(Of String)
        Dim o = ""
        Dim tmp(4) As Byte
        Dim count As Long


        While True
            tmp(3) = tcpStream.ReadByte()
            Select Case tmp(3)
```

```
                    Case 0
                        output.Add(o)
                        If o.Substring(0, 5) = "!done" Then
                            Exit While
                        Else
                            o = ""
                            Continue While
                        End If
                    Case Is < &H80
                        count = tmp(3)
                    Case Is < &HC0
                        count = BitConverter.ToInt32(New Byte() {tcpStream.ReadByte(), tmp(3), 0, 0}, 0) ^ &H8000
                    Case Is < &HE0
                        tmp(2) = tcpStream.ReadByte()
                        count = BitConverter.ToInt32(New Byte() {tcpStream.ReadByte(), tmp(2), tmp(3), 0}, 0) ^ &HC00000
                    Case Is < &HF0
                        tmp(2) = tcpStream.ReadByte()
                        tmp(1) = tcpStream.ReadByte()
                        count = BitConverter.ToInt32(New Byte() {tcpStream.ReadByte(), tmp(1), tmp(2), tmp(3)}, 0) ^ &HE0000000
                    Case &HF0
                        tmp(3) = tcpStream.ReadByte()
                        tmp(2) = tcpStream.ReadByte()
                        tmp(1) = tcpStream.ReadByte()
                        tmp(0) = tcpStream.ReadByte()
                        count = BitConverter.ToInt32(tmp, 0)
                    Case Else
                        Exit While       'err
                End Select

                For i = 0 To count - 1
                    o += ChrW(tcpStream.ReadByte())
                Next
            End While
            Return output
        End Function

        Function EncodeLength(ByVal l As Integer) As Byte()
            If l < &H80 Then
                Dim tmp = BitConverter.GetBytes(l)
                Return New Byte() {tmp(0)}
            ElseIf l < &H4000 Then
                Dim tmp = BitConverter.GetBytes(l Or &H8000)
                Return New Byte() {tmp(1), tmp(0)}
            ElseIf l < &H200000 Then
                Dim tmp = BitConverter.GetBytes(l Or &HC00000)
                Return New Byte() {tmp(2), tmp(1), tmp(0)}
            ElseIf l < &H10000000 Then
```

```
                Dim tmp = BitConverter.GetBytes(l Or &HE0000000)
                Return New Byte() {tmp(3), tmp(2), tmp(1), tmp(0)}
            Else
                Dim tmp = BitConverter.GetBytes(l)
                Return New Byte() {&HF0, tmp(3), tmp(2), tmp(1), tmp(0)}
            End If
        End Function
End Class
```

Example

```
Module Module1

      Sub Main()

            Dim mk = New Mikrotik("mikrotik")
            If Not mk.Login("admin", "PpAaSsWwOoRrDd") Then
                  Console.WriteLine("Cant log in")
                  mk.Close()
                  Console.ReadLine()
                  Return
            End If

            mk.Send("/system/clock/getall", True)
            For Each row In mk.Read()
                  Console.WriteLine(row)
            Next
            Console.ReadLine()
      End Sub

End Module
```

RouterOS PHP class

- Author: Kamil Trzcinski
- E-mail: ayufan(at)osk-net(dot)pl
- WWW: [1]
- License: GPL
- added callbacks
- added btest
- initial release

The main purpose of another RouterOS PHP API class it to simplify configuration update processes. Example: We have about 20 access points and for each of them we have connected about 20 wds links. Using automatic configuration process we can store information about all wds links in one place. It can be MySQL database.

Using set of configuration files router's can be divided into function groups (ie. router, main-access-point, client-access-point, switch) and be configured from central server automatically. ONLY changed configuration will be updated, so in most cases no configuration will change.

Requires a very good knowledge of RouterOS configuration tree, PHP Runtime and API access to RouterOS.

Base class for handing RouterOS API interface. It implements methods of getting and setting values as well restarting router.

All commands accepts two forms of arguments. Either using string or using array. Prefered way is to use **array**.

From version 0.2 interface supports many simulatenous commands using user callbacks. Every function with $callback parameter support asynchronous operation. If valid $callback would be passed function returns instead of results assigned ".tag" value to callback or FALSE on failure.

function myCallbackFunction($conn, $state, $results);

- **conn** - RouterOS object
- **state** - indicate callback boolean state. TRUE the response is either "!done" or "!re". FALSE the response is "!trap"
- **results** - contains additional arguments for response. If NULL callback got "!done" status otherwise contains associative array of results from API server.

To specify command (in RouterOS configuration tree) use:

- slash delimeted string:

```
/ip/firewall/string
```

- array of string:

```
array("ip", "firewall", "string")
```

To specify configuration line (for command) use:

- space delimeted string:

```
chain=forward action=drop in-interface=ether1
```

- associative array of string:

```
array("chain"=>"forward", "action"=>"drop", "in-interface"=>"ether1")
```

- public $readOnly = FALSE;

Read-only flag. If set to TRUE: RouterOS class will not change nor remove any item.

- static function connect($host, $login, $password, $port = 8728, $timeout = 5)

Connects to new RouterOS using specified "host" with specified "login" and "password" on "port".

```
$conn = RouterOS::connect("192.168.10.11", "admin", "adminpassword");
```

- public function setTimeout($timeout = 5)

Set socket timeout in seconds.

```
$conn->setTimeout(10);
```

- function dispatch(&$continue)

Dispatches comming messages from server to functions executed as callbacks. Returns TRUE if there is one or more pending functions.

continue - flag to manually break listener loop (it can be done from callback). Initial value should be set to TRUE.

```
$continue = TRUE; $conn->dispatch($continue);
```

- function getall($cmd, $proplist = FALSE, $args = array(), $assoc = FALSE, $callback = FALSE)

Get all values for specified command. Returns array of results.

cmd - name of command (string or array)

proplist - list of values to get (string comma delimeted or array)

args - additional arguments, ie. queries (string space delimeted or associative array)

assoc - name of associative key

```
$conn->getall("/interface/wireless/registration-table");
```

```
Array
(
    [0] => Array
        (
            [.id] => *2
            [interface] => ap11
            [mac-address] => 00:1F:1F:XX:XX:XX
            [ap] => true
            [wds] => true
            [rx-rate] => 11Mbps
            [tx-rate] => 11Mbps
            [packets] => 237069,179718
            [bytes] => 210614627,28263429
            [frames] => 237069,179718
            [frame-bytes] => 209210987,27185121
            [hw-frames] => 289168,179718
            [hw-frame-bytes] => 262600082,31498353
            [tx-frames-timed-out] => 0
            [uptime] => 1d11:00:24
            [last-activity] => 00:00:04.950
            [signal-strength] => -62dBm@1Mbps
            [signal-to-noise] => 29
            [strength-at-rates] => -62dBm@1Mbps 20ms,-61dBm@11Mbps 2m20s690ms
```

```
            [tx-ccq] => 95
            [p-throughput] => 5361
            [ack-timeout] => 30
            [last-ip] => 192.168.9.14
            [802.1x-port-enabled] => true
            [wmm-enabled] => false
        )

    [1] => Array
        (
            [.id] => *7
            [interface] => backbone
            [radio-name] => XXXX
            [mac-address] => 00:0C:42:XX:XX:XX
            [ap] => true
            [wds] => true
            [rx-rate] => 54Mbps*2
            [tx-rate] => 54Mbps*2
            [packets] => 22113864,21168612
            [bytes] => 3001775892,3956497045
            [frames] => 20116089,17752199
            [frame-bytes] => 2899204750,3906321077
            [hw-frames] => 34728036,595903321
            [hw-frame-bytes] => 4191331598,1269068004
            [tx-frames-timed-out] => 0
            [uptime] => 1d11:00:22
            [last-activity] => 00:00:00
            [signal-strength] => -62dBm@6Mbps
            [signal-to-noise] => 33
            [strength-at-rates] => -62dBm@6Mbps 0s,-61dBm@9Mbps 6m37s360ms,-63dBm@12Mbps
            [tx-signal-strength] => -59
            [tx-ccq] => 100
            [rx-ccq] => 97
            [p-throughput] => 55138
            [nstreme] => true
            [framing-mode] => best-fit
            [framing-limit] => 3200
            [routeros-version] => 4.2
            [last-ip] => 192.168.254.2
            [802.1x-port-enabled] => true
            [compression] => false
            [wmm-enabled] => true
        )
)
```

```
$conn->getall("/interface/wireless/registration-table", ".id,interface,mac-address", FALSE, "mac-address");
```

```
$conn->getall(array("interface", "wireless", "registration-table"), array(".id", "interface", "mac-address"), FALSE, "mac-address");
```

```
Array
(
    [00:1F:1F:XX:XX:XX] => Array
        (
            [.id] => *2
            [interface] => ap11
            [mac-address] => 00:1F:1F:XX:XX:XX
        )

    [00:0C:42:XX:XX:XX] => Array
        (
            [.id] => *7
            [interface] => backbone
            [mac-address] => 00:0C:42:XX:XX:XX
        )
)
```

- function set($cmd, $args, $callback = FALSE)

Set item or command value.

```
$conn->set("/ip/firewall/filter", array(".id"=>"*10", "chain"=>"forward", "action"=>"reject");
```

- function reboot()

Reboots RouterOS. Returns TRUE on success.

- function cancel($tag = FALSE, $callback = FALSE)

Cancel last or tagged command. Returns TRUE on success.

- function fetchurl($url, $callback = FALSE)

Uses /tool/fetch to download file from remote server. It can be used for example to fetch latest RouterOS releases. Returns TRUE on success.

```
$conn->fetchurl("http://66.228.113.58/routeros-mipsbe-4.3.npk");
```

- function move($cmd, $id, $before, $callback = FALSE)

Move specified item before another item. Returns TRUE on success.

```
$conn->move("/ip/firewall/filter", "*5", "*10");
```

- function add($cmd, $args, $callback = FALSE)

Add new item for command. Returns new ID on success.

```
$conn->add("/ip/firewall/filter", "chain=forward action=drop");
```

- function remove($cmd, $id, $callback = FALSE)

Remove specified item or array of items for command. Returns TRUE on success.

```
$conn->remove("/ip/firewall/filter", "*10");
```

```
$conn->remove("/ip/firewall/filter", array("*10", "*20"));
```

- function unsett($cmd, $id, $value, $callback = FALSE)

Unset value for specified item. Returns TRUE on success.

```
$conn->unsett("/queue/simple", "*10", "time");
```

- function btest($address, $speed = "1M", $protocol = "tcp", $callback = FALSE)

Perform a bandwidth-test. Supports only transmit and it should be used as asynchronous command, ie. callback.

- function scan($id, $duration="00:02:00", $callback = FALSE)

Perform a remote wireless scan. Before scanning set stream interval to larger value than duration. Returns array of results on success.

```
$interfaces = $conn->getall("/interface/wireless", ".id,name", FALSE, "name");
```

```
Array
(
    [bridge06] => Array
        (
            [.id] => *9
            [name] => bridge06
        )

    [backbone] => Array
        (
            [.id] => *A
            [name] => backbone
        )
)
```

```
$results = $conn->scan($interfaces["backbone"][".id"]);
```

```
Array
(
    [00:02:6F:XX:XX:XX] => Array
        (
            [address] => 00:02:6F:XX:XX:XX
            [ssid] => bridge02
            [band] => 5ghz-t
            [freq] => 5210
            [sig] => -58
            [nf] => -105
            [snr] => 47
            [radio-name] => 1402
        )

    [00:0C:42:XX:XX:XX] => Array
        (
            [address] => 00:0C:42:XX:XX:XX
            [ssid] => bridge13
            [band] => 5ghz-t
            [freq] => 5210
            [sig] => -60
            [nf] => -105
```

```
                [snr] => 45
                [radio-name] => 3713
            )
)
```

```
<?

require_once(dirname(__FILE__)."/routeros.class.php");

if($argc < 3) {
  die("usage: ${argv[0]} <login>:<password>@<host> <destination1>@<speed>@<protocol>...\n");
}

// get args
list($login, $host) = explode('@', $argv[1], 2);
if($host) {
  list($login, $password) = explode(':', $login, 2);
}
else {
  $host = $login;
  $login = "admin";
  $password = "";
}

// connect to server
$conn = RouterOS::connect($host, $login, $password) or die("couldn't connect to $login@$host\n");
$conn->setTimeout(60);

// structures
$dests = array();
$status = array();
$current = array();
$average = array();
$percent = array();
$tags = array();

// start btest
for($i = 2; $i < $argc; ++$i) {
  list($dest, $speed, $protocol) = explode("@", $argv[$i]);

  if(!$speed)
    $speed = 0;
  if(!$protocol)
    $protocol = "tcp";

  $name = gethostbynamel($dest);
  if($name === FALSE)
      die("couldn't resolve $dest!\n");
```

```
  $name = $name[0];

  if($dests[$name])
    die("destination $dest already defined!\n");

  $tag = $conn->btest($name, $speed, $protocol, btestCallback);
  if($tag === FALSE)
    continue;

  $tags[$tag] = $name;
  $dests[$name] = array("dest" => $dest, "speed" => $speed, "protocol" => $protocol);
}

// print header
ncurses_init();
ncurses_nl();
printStatus();

// dispatch messages
$continue = TRUE;
$conn->dispatch($continue);

exit;

function btestCallback($conn, $state, $results) {
  global $dests, $tags, $status, $current, $average, $percent;

  // done message
  if($state == TRUE && !$results)
    return;

  // find destination
  $dest = $tags[$results[".tag"]];
  if($dest === FALSE)
    return;

  // trap message
  if($state == FALSE) {
    if($results["message"] == "interrupted")
      return;

    // state changed
    if($status[$dest] != $results["message"]) {
      $status[$dest] = $results["message"];
      printStatus();
    }
    return;
```

```
  }

  // not running
  if($results["status"] != "running") {
    // state changed
    if($status[$dest] != $results["status"]) {
      $status[$dest] = $results["status"];
      printStatus();
    }

    // restart btest (in error state)
    if($results["status"] != "connecting") {
      $conn->cancel($results[".tag"]);
      $tag = $conn->btest($dest, $dests[$dest]["speed"], $dests[$dest]["protocol"], btestCallback);
      if($tag !== FALSE)
        $tags[$tag] = $dest;
    }
    return;
  }

  // running get results
  $status[$dest] = $results["status"];
  $current[$dest] = bytesToString($results["tx-current"], 1000, "b");
  $average[$dest] = bytesToString($results["tx-10-second-average"], 1000, "b");
  $percent[$dest] = round(100 * $results["tx-10-second-average"] / stringToBytes($dests[$dest]["speed"], 1000), 1);
  printStatus();
}

function stringToBytes($data, $multi = 1024) {
  $value = floatval($data);
  switch(substr(strtolower($data), -1)) {
    case 'g':
      $value *= $multi;
    case 'm':
      $value *= $multi;
    case 'k':
      $value *= $multi;
  }
  return $value;
}

function bytesToString($data, $multi = 1024, $postfix = "B") {
  $data = intval($data);

  if($data < $multi) {
    return round($data, 0) . $postfix;
  }
```

```
  if($data < $multi*$multi) {
    return round($data/$multi, 1) . "k$postfix";
  }
  if($data < $multi*$multi*$multi) {
    return round($data/$multi/$multi, 1) . "M$postfix";
  }
  return round($dat /$multi/$multi/$multi, 1) . "G$postfix";
}

function getTime() {
  static $startTime;
  if(!$startTime)
    $startTime = microtime(TRUE);
  return round(microtime(TRUE) - $startTime, 1);
}

function printTable($header, $line) {
  $sizes = array();
  foreach($header as $h)
    $sizes[$h] = strlen($h);

  foreach($line as $v)
    foreach($header as $h)
      $sizes[$h] = max($sizes[$h], strlen($v[$h]));

  $out = "== ";
  foreach($header as $h)
    $out .= str_pad($h, $sizes[$h])." == ";
  $out .= "\n";

  foreach($line as $v) {
    $out .= "-- ";
    foreach($header as $h)
      $out .= str_pad($v[$h], $sizes[$h])." -- ";
    $out .= "\n";
  }
  return $out;
}

function printStatus() {
  global $dests, $status, $current, $average, $percent;

  ncurses_clear();
  ncurses_move(0, 0);
  ncurses_addstr("time: ".getTime()."\n\n");

  $header = array("host", "speed", "proto", "status", "current", "average", "%");
```

```
  $lines = array();


  foreach($dests as $dest=>$desc) {
    $lines[] = array("host"=>$desc["dest"], "speed"=>$desc["speed"], "proto"=>$desc["protocol"],
"status"=>$status[$dest], "current"=>$current[$dest], "average"=>$average[$dest], "%"=>$percent[$dest]);
  }
  ncurses_addstr(printTable($header, $lines));
  ncurses_refresh();
}


?>
```

Parser class to load configuration from file and perform differencing configuration update.

Parser output should be shown in text/plain content-type!

Order of sections defines order of configuration updates.

- For each section: **getall** items from RouterOS
- ignore all dynamic entries, remove all invalid entries
- try to classify RouterOS item to either ignore or to pass list
- try to match RouterOS item with local item using defined **keys**, if no match found remove, if match found update only what changed
- reorder RouterOS item list
- add not found items to RouterOS

One line is one command. Command can be either: comment, flow function, include function, user function definer or configurer function. Each line is firstly trimmed from whitespaces.

Comment can only by started from new line and after # char. Before # can be zero or more whitespaces. **Comment can't be added after command!**

```
# Sample comment
```

To use variable add % before and after variable name: To get value of **my-variable**:

```
%my-variable%
```

To define variable in script use **var** keyword

```
var [variable-name] [variable-value]
```

```
var my-variable test-variable
```

```
var my-variable test-variable-using-previous-value-of-my-variable-%my-variable%
```

In script files there are a few flow functions: if, elseif, else, endif. Flow functions can be nested. One flow block if-else-endif has to be located in ONE file. So simply You can't start flow block in one file and end it in another.

```
 if [left-value] [operator] [right-value]
# [commands execute when first comparision is true] ...
elseif [left-value] [operator] [right-value]
# [commands execute when second comparision is true] ...
else
# [commands execute when neither first nor second comparision is true] ...
endif
```

left-value, right-value - either string or variable

```
if %version% ~= 4.*
# execute commands for version 4.*
else
# execute commands for all other versions 4.*
endif
```

- **=** - left is equal right
- **!=** - left is not equal tight
- **<** - left is less than right
- **<=** - left is less or equal right
- **>** - left is greater than right
- **>=** - left is greater or equal right
- **~=** - right is wildcardly equal left (using fnmatch from php)
- **!~=** - right is wildcardly not equal left (using fnmatch from php)

Script file can include another file and parse it in place. Current file path is used to include file. Each file can be included more than once.

```
include [file-name]
```

Includes file if exists. If file doesn't exist parser will continue.

```
require [file-name]
```

Require file. If file doesn't exist parser will return with error.

```
set [alias] [key=value] [key2=value2]...
```

```
add [alias] [key=value] [key2=value2]...
```

Set or add (synonims) config line.

```
function [cmd-name] [$arg1] [$arg2=$default2]...
# function body
endfunction
```

Add new user php function. Always in defined function first argument is $parser to access current parser context.

```
function my_first_function $srcaddress $dstaddress="1.2.3.4"
# do some crazy stuff with $parser.
endfunction
```

```
ignore [alias] [key=value] [key2=value2]...
```

```
pass [alias] [key=value] [key2=value2]...
```

Add ignore or pass config line. See RouterOSParser::ignore or RouterOSParser::pass functions.

```
flush [alias] [alias2]...
```

Clean all configuration for specified alias!

```
section [alias] [cmd] [type] [keys or false] [default_key=default_value]...
```

Add new section **alias** of **type** to configuration update with comma delimeted group **keys** and list of **default_key**. See RouterOSParser::section function.

```
section firewall-filter /ip/firewall/filter addset_order
```

```
section wireless-wds /interface/wireless/wds addset name disabled=no
```

disable [alias] [alias2]...

Remove section from configuration update.

```
disable firewall-filter queue-tree
```

[cmd-name] [arg1] [arg2]...

Execute user defined function with args.

```
my_first_function 192.168.10.1 192.168.10.254
```

To perform automatic update updater has to know what type of data to expect. Proper section type has to be specified.

- **addset** - add, set or remove items (where .id is specified for an item) in unspecified order (ie. /queue/tree, /queue/types...)
- **addset_order** - add, set or remove items where order of items matters(ie. /ip/firewall/filter, /queue/simple, /ip/dns/static...)
- **set** - only set items (where .id is specified for an item), don't remove or add an new one (ie. /interface, /queue/interface...)
- **value** - only set variables (where are values not items) (ie. /ip/firewall/connection/tracking, /ip/dns)
- public $logs = array();

Array of logs from section update.

- public $showIgnored = FALSE;

Whatever to show in logs items which found to be "ignore" or "pass".

- public $currentContext;

Information about current processed line (can be accessed from user php functions).

- function error($message)

DIE execution with specified error message.

- function define($key, $value = FALSE)

Define or undefine script variable: %key%

- function variable($key)

Get value of script variable: %key%

- function replace($value)

Replace string using script variables

```
$parser->define('var', 'test-of-var');
```

```
$parser->replace('string of variable: %var%');
```

```
"string of variable: test-of-var"
```

- function config($cmd, $line)

Add or set config for specified short command.

```
$parser->config("firewall-filter", "action=drop chain=forward");
```

```
$parser->config("firewall-filter", array("action"=>"drop", "chain"=>"forward"));
```

```
$parser->config("connection-tracking", "enabled=no");
```

- function ignore($cmd, $line)

Ignore specified item from synchronization. Has precedence before "pass". Muliple ignore or pass rules can be added.

```
$parser->ignore("firewall-filter", "chain=forward"); // doesn't synchronize rules from chain "forward"
```

- function pass($cmd, $line)

Pass only specified item to synchronization. Muliple ignore or pass rules can be added.

```
$parser->pass("firewall-filter", "chain=forward"); // synchronize ONLY rules from chain "forward"
```

- function section($alias, $cmd, $type, $keys = FALSE, $defaults = FALSE)

Add a section to synchronization.

alias - short alias of section

cmd - RouterOS command

type - type of section. See Types of section.

keys - list of keys to perform differencing synchronization.

defaults - list of default values.

```
$parser->section("firewall-filter", "/ip/firewall/filter", "addset_order", FALSE, "disabled=no");
    // ordered list of items without <b>key</b> and default not disabled
```

```
$parser->section("wireless-wds", "/interface/wireless/wds", "addset", "name", "disabled=no");
    // unordered list of items with interface name as key and default not disabled
```

```
$parser->section("interface-queue", "/queue/interface", "set", "name");
    // only settable list of items with interface name as key
```

```
$parser->section("dns", "/ip/dns", "value");
    // value section
```

- function cmd($alias, $cmd)

Defines user php function with at least one arg: **$parser**.

```
function parser_test_function($parser, $srcaddress, $dstaddress) {
  $parser->config("firewall-filter", "chain=forward action=drop src-address=$srcaddress dst-address=$dstaddress");
  return TRUE;
}
$parser->cmd("test_function", parser_test_function);
```

- function parseFile($file)

Parse text from file. If $file is array parse line of array as command.

```
$parser->parseFile("my_config_file.cfg");
```

```
$parser->parseFile(array("config line 1", "config line 2", "config line 3"));
```

- function call($cmd, $args)

Execute defined function with specified args as array in current parser context. Returns function return value.

```
$parser->call("test_function", array("192.168.10.1", "1.2.3.4"));
```

- function updateSection($conn, $alias)

Perform specified section $alias update for specified RouterOS $conn connection. All update logs are in $parser->logs.

```
$parser->updateSection($conn, "firewall-filter");
```

- function update($conn, $ret = FALSE)

Perform update of all sections for specified RouterOS $conn connection. All logs are either flushed on stdout or returned if $ret is TRUE.

```
$parser->update($conn);
```

```
<?


require_once("routeros.class.php");
require_once("routerosparser.class.php");


header("Content-Type: text/plain");


// connect to device
$conn = RouterOS::connect("192.168.10.11", "admin", "adminpassword") or die("couldn't connect to 192.168.10.11");
$resource = $conn->getall(array("system", "resource")) or die("couldn't get resource;


// create class and define device information
$parser = new RouterOSParser();
$parser->variable("name", "MikroTik");
$parser->variable("version", $resource["version"]);
$parser->variable("arch", $resource["architecture-name"]);


// define function
function allowConnectivity($parser, $srcaddress, $dstaddress) {
  if(!$srcaddress || !$dstaddress)
    $parser->error("src or dst not specified");
  $parser->config("firewall-filter", "chain=forward src-address=$srcaddress dst-address=$dstaddress action=accept");
  $parser->config("firewall-filter", "chain=forward src-address=$dstaddress dst-address=$srcaddress action=accept");
}
$parser->cmd('allow-forward', allowConnectivity);


$parser->section("firewall-filter", "/ip/firewall/filter", "addset_order"); // add firewall-filter
$parser->pass("firewall-filter", "chain=forward"); // update ONLY forward chain


$parser->parseFile("example_config.cfg"); // load configuration
```

```
$parser->update($conn); // perform update


?>
```

```
# load predefined global OSPF configuration
require ospf.cfg

# set device name, clock and ntp-client
section identity /system/identity value
section clock /system/clock value
section ntp-client /system/ntp/client value
set identity name=%name%
set clock time-zone-name=Europe/Warsaw
set ntp-client enabled=true mode=unicast primary-ntp=192.168.10.5 secondary-ntp=192.168.10.6

# of course in later part of configuration you can change configuration
set ntp-client enabled=false

# custom commands
add firewall-filter in-interface=ether1 out-interface=ether2 chain=forward action=drop

# allow forward for a few clients
allow-forward 192.168.1.1 192.168.10.5
allow-forward 192.168.1.5 192.168.10.5
allow-forward 192.168.1.10 192.168.10.5

# define your own function
function allow-interface $interface
  if(!$interface)
    $parser->error("interface not specified");
  $parser->config("firewall-filter", "in-interface=$interface chain=forward action=accept");
  $parser->config("firewall-filter", "out-interface=$interface chain=forward action=accept");
endfunction

allow-interface ether1
allow-interface ether2
```

```
# check RouterOS version and configure differently
if %version% ~= 3.*
        section ospf /routing/ospf value
        set ospf distribute-default=never
redistribute-connected=as-type-2 redistribute-static=as-type-2
redistribute-rip=no redistribute-bgp=no metric-default=2
metric-connected=2 metric-static=1 metric-rip=1 metric-bgp=1
else
        section ospf-instance /routing/ospf/instance addset name
        add ospf-instance name=default distribute-default=never
redistribute-connected=as-type-2 redistribute-static=as-type-2
```

```
redistribute-rip=no redistribute-bgp=no metric-default=2
metric-connected=2 metric-static=1 metric-rip=1 metric-bgp=1
endif

# add sections for ospf configuration
section ospf-area /routing/ospf/area addset name,area-id
section ospf-interface /routing/ospf/interface addset interface
section ospf-network /routing/ospf/network addset network
section ospf-area-range /routing/ospf/area/range addset area,range

# add configuration
add ospf-area name=backbone area-id=0.0.0.0 type=default disabled=false
add ospf-interface interface=private cost=10 priority=10
authentication=md5 authentication-key=MagicPassword
network-type=broadcast retransmit-interval=00:00:10
transmit-delay=00:00:04 hello-interval=00:00:20 dead-interval=00:01:00
add ospf-network network=192.168.10.0/24 area=backbone disabled=false
```

- License [2]
- SVN (login: guest) [3]
- Documentation [4]
- routeros.class.php [5]
- routerosparser.class.php [6]

References

[1] http://www.ayufan.eu
[2] http://www.gnu.org/licenses/gpl.html
[3] https://svn.osk-net.pl:444/rosapi
[4] http://ayufan.eu/src/rosapi/trunk/documentation.html
[5] http://ayufan.eu/src/rosapi/trunk/routeros.class.php
[6] http://ayufan.eu/src/rosapi/trunk/routerosparser.class.php

API command notes

Summary

This page contains some information about details of API commands, examples or use-cases. For more detailed information refer to API.

Note: Till version 4.6 including API logins where shown as winbox logins. Since 4.7 this behaviour is changed and API logins will be correctly recognised and displayed as API logins.

General information about API sentences

Communication with router through API is done using **API sentences** that consist of **API command** and **attributes**. API queries are considered special command attribute, for example **command tags**. In each sentence can only be **one command** and *'many attributes*.

Command

API command is command as it is available from CLI (or special API command like 'getall'). Command syntax is derived from CLI and includes CLI path to and command itself.

For example:

```
/ip address print
```

API command derived from this CLI command will be:

```
/ip/address/print
```

in this case, /ip/address/ is path and print is command itself, but, since *print* or command on its own does not have meaning, path+command is considered to be command as that determines what to do exactly.

Attributes

CLI Attributes

Each API sentence can have attributes. Full attribute list can be acquired from CLI using *?* or double *Tab* key.

Example:

First, what command we are going to use? In CLI we will examine /ip address add command.

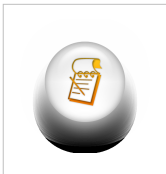
What attributes command has? result of *?*

```
[admin@MikroTik] > ip address add
Creates new item with specified property values.

address -- Local IP address
broadcast -- Broadcast address
comment -- Short description of the item
copy-from -- Item number
disabled -- Defines whether item is ignored or used
interface -- Interface name
netmask -- Network mask
network -- Network prefix
```

result of double *Tab*

```
[admin@MikroTik] > ip address add
broadcast   comment   copy-from   disabled   netmask   network   address   interface
```



Note: Not all attributes will have full or precise description in CLI, but all attributes will have precise and full description of values accepted by attribute

Building API sentence:

```
/ip/address/add
=address=192.168.88.1/24
=interface=ether1
```

Result of execution of this command will be IP address added on interface ether1 same as in CLI.

Note: If command in CLI does not have named attribute using *?* key you can get required attribute name. Atribute that is not named will appear in between <>

API attributes

API has some special attributes, that are not available through CLI, or are not available through CLI directly. These atributes starts with dot. For example .id, that gives identification number of the item, whilst these can be seen in CLI (returned by find) ID is not directly shown with the item.

API command attributes

All attributes of command starts with equals sigh, whilst special case for API attribute to command itself. Like **.tag** attribute that is not part of any command, but can be used to identify returned data of command executed.

Atributes without value

Commands in RouterOS have attributes that does not have any value set, If these attributes are used it just indicates that they should be used, and value, if any is given will be ignored.

For example, indicate that we will follow IP address changes:

```
/ip/address/print
=follow=
```

See the equals marks surrounding *follow* - they should be there as attribute should be between them.

API sentence structure

API sentence should be sent in very specific form. About precise descriptions please see API. If you are not going to write your own API implementations or, you do not understand exactly how it should be created, here is the explanation:

- API sentence can consist of several lines (or words);
- when sent to router each word have to have a prefix, that have to be made in a specific way encoding length of the word;
- last word in API sentence have to be zero terminated (have to contain byte set to all zeros). Also, if sentence only contains one word, it has to be zero terminated, or else router will wait for further words in that sentence, and all other words will be counted as words from same sentence, not new sentence.

All this boils down to this, where *XX* is encoded word legth, *aaaa* is word and *0x00* is terminating zero single line sentence

```
XXaaaa0x00
```

multiple line sentence

```
XXaaaa
XXaaaa0x00
```

or

```
XXaaaa
XXaaaa
XXaaaa0x00
```

Note: Usually API implementations takes care of *encoding word length* part and user only have to worry to make sure that correct method/function is used to send words over to router and make sure words make meaningful sentence for execution

Scripting and API

It is possible to access RouterOS scripting global variables through the API if user have enough permissions to read this menu.

```
/system/script/environment
```

Users are able to remove or alter value of the variable. Keep in mind, that variable type is automatically determined by scripting engine. Be aware that variable type can change while you are working with it.

Also, no other scripting constructs are available in API (:if, :for etc.)

Note: Through API it is not possible to create new variables

Note: Find command have many constructs that are part of scripting, thus not available through API

API login

since RouterOS 4.7 it is possible to monitor all API connections to RouterOS under */user active* menu in console (or corresponding menu in winbox). Same way that can be done for telnet, ssh, winbox and webfig logins.

!fatal

!fatal can be received only in cases when API is closing connection:

- too many commands are sent to router prior login
- there is error in authentication that is not recoerable
- **/quit** command is sent to router. Response looks like this:

```
>>> /quit

<<< !fatal
<<< session terminated on request
```

CLI commands that are not in API

some commands are not available in API when compared with CLI, these include interactive commands and scripting commands

Interactive commands

interactive command examples that will not work in API are:

```
/system telnet
```

```
/system ssh
```

```
/tool mac-telnet
```

Scripting commands

Any find command is not supported, use queries and proplist instead

```
/ip adddress find
```

Commands that starts with semicolon:

```
delay  error    find  foreach  if   local    parse
put  set   toarray  toid  toip6  tostr   typeof
do     execute  for   global   len  nothing
pick   resolve  time  tobool   toip  tonum  totime  while
```

API sentence examples

Examples of use of commands

Addressing entries

In some places in API it is possible to address entries using value of *name* attribute as attribute *.id* value. Some places where ambiguity could arise this feature is not available.

Examples

setting interface name to one that already exist:

```
/interface/set
=.id=ether1
=name=ether2
```

will result in:

```
!trap
=category=4
=message=already have device with such name

!done
```

While adding several entries with same name as static DNS entries is completely legal, addressing entries using *name* value for .id is **NOT**. Entry with name=example.com address=192.168.88.1 added before.

```
/ip/dns/static/set
=.id=example.com
=address=3.3.3.3
```

The result

```
!trap
=category=0
=message=no such item
```

Monitor-traffic

it is equivalent of CLI */interface monitor-traffic* command

Details

- Basic command syntax:

```
/interface/monitor-traffic
=interface=<id1>,<id2>,<id3>
```

- Output: replies will be sent in succession with in statistics about interface in order of IDs given in command. So, first re! will be for <id1>, second for <id2>
- Duration: command runs until interrupted with /cancel
- planned changes: it is planned to add item identification to replies.
- since interfaces have name field, value from that field can be used to address interface instead of .id

Example

- Command

```
/interface/monitor-traffic
=interface=ether1-Local,ether3-Out
```

- Return

```
!re
=rx-packets-per-second=4
=rx-drops-per-second=0
=rx-errors-per-second=0
=rx-bits-per-second=8531
=tx-packets-per-second=3
=tx-drops-per-second=0
=tx-errors-per-second=0
```

```
=tx-bits-per-second=11266

!re
=rx-packets-per-second=8
=rx-drops-per-second=0
=rx-errors-per-second=0
=rx-bits-per-second=14179
=tx-packets-per-second=4
=tx-drops-per-second=0
=tx-errors-per-second=0
=tx-bits-per-second=8591

!re
=rx-packets-per-second=4
=rx-drops-per-second=0
=rx-errors-per-second=0
=rx-bits-per-second=2312
=tx-packets-per-second=2
=tx-drops-per-second=0
=tx-errors-per-second=0
=tx-bits-per-second=3039

!re
=rx-packets-per-second=5
=rx-drops-per-second=0
=rx-errors-per-second=0
=rx-bits-per-second=4217
=tx-packets-per-second=1
=tx-drops-per-second=0
=tx-errors-per-second=0
=tx-bits-per-second=635
```

Ping v4.x and older

it is not equivalent of ping available in CLI, but it supports same arguments and working principles are the same. Only difference is in data returned.

Details

- ping in API reports how many successful replies it has received. And can only be used to determine if target host is capable of replying to ICMP requests
- for ease of use it us suggested that it is used with count argument set to some value
- Ping returns only when it is interrupted or reached count limit.

Example

```
/ping
=address=192.168.88.1
=count=3
```

In this case ping returned after duration*count seconds, where duration was default 1 second.

```
!done
=ret=3
```

Ping v5.x and newer

it is equivalent of ping available in CLI, but it will give report on averages every time it has result for sent ping.

Details

- for ease of use it us suggested that it is used with count argument set to some value
- Ping returns only when it is interrupted or reached count limit.
- Timing results are in form HH:MM:SS.sss (HH - hours; MM - minutes; SS - seconds; sss - miliseconds)

Example

```
/ping
=address=192.168.88.1
=count=2
```

In this case ping returned after duration*count seconds, where duration was default 1 second.

```
!re
=host=192.168.88.1
=size=56
=ttl=42
=time=00:00:00.001
=sent=1
=received=1
=packet-loss=0
=min-rtt=00:00:00.001
=avg-rtt=00:00:00.001
=max-rtt=00:00:00.001

!re
=host=192.168.88.1
=size=56
=ttl=42
=time=00:00:00.001
=sent=2
=received=2
=packet-loss=0
=min-rtt=00:00:00.001
=avg-rtt=00:00:00.001
=max-rtt=00:00:00.001
```

```
!done
```

Cancel tagging

You may cancel every previously executed task. Note however how cancel behaves with specified tag and without.

Example without tag

Command execution.

```
/ping
=address=google.com
```

Reply itself.

```
!re
=host=77.252.2.103
=size=56
=ttl=60
=time=00:00:00.022
=sent=1
=received=1
=packet-loss=0
=min-rtt=00:00:00.022
=avg-rtt=00:00:00.022
=max-rtt=00:00:00.022

!re
=host=77.252.2.103
=size=56
=ttl=60
=time=00:00:00.026
=sent=2
=received=2
=packet-loss=0
=min-rtt=00:00:00.022
=avg-rtt=00:00:00.024
=max-rtt=00:00:00.026
```

Cancel reply. Notice 2 !done words.

```
/cancel

!trap
=category=2
=message=interrupted

!done

!done
```

Example with specified tag

Command execution.

```
/ping
=address=google.com
.tag=22
```

Reply itself.

```
!re
=host=5.226.127.144
=size=56
=ttl=61
=time=00:00:00.008
=sent=1
=received=1
=packet-loss=0
=min-rtt=00:00:00.008
=avg-rtt=00:00:00.008
=max-rtt=00:00:00.008
.tag=22

!re
=host=5.226.127.144
=size=56
=ttl=61
=time=00:00:00.025
=sent=2
=received=2
=packet-loss=0
=min-rtt=00:00:00.008
=avg-rtt=00:00:00.016
=max-rtt=00:00:00.025
.tag=22
```

Cancel command.

```
/cancel
=tag=22
.tag=1

!trap
=category=2
=message=interrupted
.tag=22

!done
.tag=1

!done
```

```
.tag=22
```

Canceling with additional errors

Cancel failed /tool fetch via http. 3 is that tag of /tool/fetch, 7 is a /cancel tag itself.

```
/cancel
=tag=3
.tag=7

!trap
=category=2
=message=interrupted
.tag=3

!done
.tag=7

!trap
=message=failure: 301 Moved Permanently
.tag=3

!done
.tag=3
```

Conclusions

As you can see in above examples, tagging sentences lets you easilly determine which command have finished. Please note that every time you cancel

```
!trap
=category=2
=message=interrupted
```

is generated.

API Ruby class

Ruby GEM

The API Ruby class(es) are now packaged together as a Ruby GEM. The latest GEM is available for download from the author's web site. The current version is 4.0.0 available here:

mtik-4.0.0.gem [1]

Or you can simply do:

```
gem install mtik
```

RDoc Documentation

The author's site also hosts Ruby RDoc documents for the classes implementing this API. The link is:

http://www.aarongifford.com/computers/mtik/latest/doc/ [2]

Examples

Several example scripts are included with the GEM or available for direct download from the author.

- tikcli [3] - A command-line-like interactive ruby script. You can type MikroTik API commands and arguments directly and have them executed.
- tikcommand [4] - A non-interactive command-line script to execute a single MikroTik API command and return the results to STDOUT.
- tikfetch [5] - This non-interactive command-line script lets one instruct a MikroTik device to download one or more files from the provided URL(s).
- tikjson.rb [6] - Another non-interactive command-line script that executes a single MikroTik API command and returns the results to STDOUT, however the results are encoded in JSON format. One could easily add CGI handling to this script, install it on a web server, and use it via a web browser. (The author in fact has done something like this for a JavaScript-based management web application that interacts with MikroTik devices via the API.)

Interactive examples using gem-supplied utility scripts

Here is are several example runs of the **tikcli** utility script included in the gem:

```
user@bsdhost:~$ tikcli 10.20.30.1 admin wrongpassword
<<< '/login' (6)
<<< END-OF-SENTENCE

>>> '!done' (5)
>>> 'ret=bf41fd4286417870c5eb86674a3b8fe4' (36)
>>> '.tag=0' (6)
>>> END-OF SENTENCE

<<< '/login' (6)
<<< '=name=admin' (11)
<<< '=response=0003a042937d84ca4bc4cf7da50aadd507' (44)
```

```
<<< END-OF-SENTENCE

>>> '!trap' (5)
>>> 'message=cannot log in' (21)
>>> '.tag=1' (6)
>>> END-OF SENTENCE

>>> '!done' (5)
>>> '.tag=1' (6)
>>> END-OF SENTENCE

=== LOGIN ERROR: Login failed: cannot log in
user@bsdhost:~$
```

That run was deliberately with the wrong password. Here's the login with the correct password:

```
user@bsdhost:~$ tikcli 10.20.30.1 admin correctpassword
<<< '/login' (6)
<<< END-OF-SENTENCE

>>> '!done' (5)
>>> 'ret=857e91c460620a02c3ca72ea7cf6c696' (36)
>>> '.tag=0' (6)
>>> END-OF SENTENCE

<<< '/login' (6)
<<< '=name=admin' (11)
<<< '=response=001a77aec14077ec267c5297969ba1fa24' (44)
<<< END-OF-SENTENCE

>>> '!done' (5)
>>> '.tag=1' (6)
>>> END-OF SENTENCE



Command (/quit to end):
```

At this point, the interactive client will accept MikroTik API commands in the format `/command/name arg1 arg2 arg3` or also `12:/command/name arg1 arg2 arg3` where the **12:** is a custom numeric prefix that tells the Ruby interactive client to auto-cancel the command in question after exactly **12** reply sentences are received, since otherwise a command with continuous output would hang the single-threaded interactive client in an endless reply-handling loop (until someone aborted it).

Arguments to API commands in this interactive client must ALREADY be in the API argument form. For example:

```
Command (/quit to end): /interface/getall ?name=ether1
=== COMMAND: /interface/getall ?name=ether1
<<< '/interface/getall' (17)
<<< '?name=ether1' (12)
<<< END-OF-SENTENCE
```

```
>>> '!re' (3)
>>> '.id=*5' (6)
>>> 'name=ether1' (11)
>>> 'type=ether' (10)
>>> 'mtu=1500' (8)
>>> 'l2mtu=1500' (10)
>>> 'bytes=26908361008/15001379552' (29)
>>> 'packets=34880279/26382227' (25)
>>> 'drops=0/0' (9)
>>> 'errors=5/0' (10)
>>> 'dynamic=false' (13)
>>> 'running=true' (12)
>>> 'disabled=false' (14)
>>> 'comment=' (8)
>>> '.tag=4' (6)
>>> END-OF SENTENCE

>>> '!done' (5)
>>> '.tag=4' (6)
>>> END-OF SENTENCE


Command (/quit to end):
```

Did you see how the user properly prefixed the query parameter `name` with the query character (question mark **?**) and also paired it via **=** with the query value? With this example CLI, you must manually format all arguments as specified by the MikroTik API.

You may have noticed that this Ruby API implementation automatically adds a unique `.tag` to every command. That means if you specify a tag value, the Ruby code will ignore it and use its own. It adds a tag so that replies can be correctly matched to the appropriate request.

Now here's the same query again, only add another parameter, `=interval=1` so that the command will repeatedly send output each second. To avoid the command continuing forever, it will be prefixed with **6:** (this CLI script strips the digit(s) and colon before sending the command to the device) to limit the number of response sentences to exactly six before the interactive client will automagically issue an appropriate `/cancel =tag=XYZ` command to cancel it.

```
Command (/quit to end): 6:/interface/getall ?name=ether1 =interval=1
=== COMMAND: /interface/getall ?name=ether1 =interval=1
<<< '/interface/getall' (17)
<<< '?name=ether1' (12)
<<< '=interval=1' (11)
<<< END-OF-SENTENCE

>>> '!re' (3)
>>> '.id=*5' (6)
>>> 'name=ether1' (11)
>>> 'type=ether' (10)
```

```
>>> 'mtu=1524' (8)
>>> 'l2mtu=1524' (10)
>>> 'bytes=26909135851/15002882324' (29)
>>> 'packets=34886461/26387909' (25)
>>> 'drops=0/0' (9)
>>> 'errors=5/0' (10)
>>> 'dynamic=false' (13)
>>> 'running=true' (12)
>>> 'disabled=false' (14)
>>> 'comment=' (8)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> '.id=*5' (6)
>>> 'name=ether1' (11)
>>> 'type=ether' (10)
>>> 'mtu=1524' (8)
>>> 'l2mtu=1524' (10)
>>> 'bytes=26909140098/15002892177' (29)
>>> 'packets=34886498/26387943' (25)
>>> 'drops=0/0' (9)
>>> 'errors=5/0' (10)
>>> 'dynamic=false' (13)
>>> 'running=true' (12)
>>> 'disabled=false' (14)
>>> 'comment=' (8)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> '.id=*5' (6)
>>> 'name=ether1' (11)
>>> 'type=ether' (10)
>>> 'mtu=1524' (8)
>>> 'l2mtu=1524' (10)
>>> 'bytes=26909141508/15002893670' (29)
>>> 'packets=34886508/26387951' (25)
>>> 'drops=0/0' (9)
>>> 'errors=5/0' (10)
>>> 'dynamic=false' (13)
>>> 'running=true' (12)
>>> 'disabled=false' (14)
>>> 'comment=' (8)
>>> '.tag=2' (6)
>>> END-OF SENTENCE
```

```
>>> '!re' (3)
>>> '.id=*5' (6)
>>> 'name=ether1' (11)
>>> 'type=ether' (10)
>>> 'mtu=1524' (8)
>>> 'l2mtu=1524' (10)
>>> 'bytes=26909143624/15002895110' (29)
>>> 'packets=34886524/26387963' (25)
>>> 'drops=0/0' (9)
>>> 'errors=5/0' (10)
>>> 'dynamic=false' (13)
>>> 'running=true' (12)
>>> 'disabled=false' (14)
>>> 'comment=' (8)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> '.id=*5' (6)
>>> 'name=ether1' (11)
>>> 'type=ether' (10)
>>> 'mtu=1524' (8)
>>> 'l2mtu=1524' (10)
>>> 'bytes=26909144116/15002895406' (29)
>>> 'packets=34886530/26387967' (25)
>>> 'drops=0/0' (9)
>>> 'errors=5/0' (10)
>>> 'dynamic=false' (13)
>>> 'running=true' (12)
>>> 'disabled=false' (14)
>>> 'comment=' (8)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> '.id=*5' (6)
>>> 'name=ether1' (11)
>>> 'type=ether' (10)
>>> 'mtu=1524' (8)
>>> 'l2mtu=1524' (10)
>>> 'bytes=26909144824/15002896659' (29)
>>> 'packets=34886535/26387973' (25)
>>> 'drops=0/0' (9)
>>> 'errors=5/0' (10)
>>> 'dynamic=false' (13)
>>> 'running=true' (12)
>>> 'disabled=false' (14)
```

```
>>> 'comment=' (8)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

<<< '/cancel' (7)
<<< '=tag=2' (6)
<<< END-OF-SENTENCE

>>> '!trap' (5)
>>> 'category=2' (10)
>>> 'message=interrupted' (19)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

=== TRAP: 'interrupted'

>>> '!done' (5)
>>> '.tag=3' (6)
>>> END-OF SENTENCE

>>> '!done' (5)
>>> '.tag=2' (6)
>>> END-OF SENTENCE


Command (/quit to end):
```

Using the Gem in Code

Execute an API command that returns endless replies until canceled

Suppose I wish to monitor traffic on an interface using the `/interface/monitor-traffic` command for 10 seconds--or in other words, because devices usually send responses to this command once every second, I want to listen for and receive 10 replies, then cancel the command:

```
<nowiki>
#!/usr/bin/env ruby

require 'rubygems'
require 'mtik'

# Be verbose in output
MTik::verbose = true

# Connect to the device:
connection = MTik::Connection.new :host => '10.0.0.1', :user => 'admin', :pass => 'password'

# We are going to send a "monitor-traffic" command that will keep sending
```

```
# output until we "cancel" the command.  We only want to receive 10 responses:
$reply_limit = 10

# Execute the command:
$reply_count = 0
connection.get_reply_each(
  "/interface/monitor-traffic",
  "=interface=ether1",
  "=.proplist=rx-bits-per-second,tx-bits-per-second"
) do |request_object, reply_sentence|
  if reply_sentence.key?('!re')  # We only pay attention to reply sentences
    # Print the reply sentence:
    p reply_sentence

    # Increment the reply counter:
    $reply_count += 1

    # If we've reached our reply goal, cancel:
    if $reply_count >= $reply_limit
      # Cancel this command request:
      request_object.cancel
    end
  end
end

connection.close
</noqiki>
```

Here's an example of output:

```
<<< '/login' (6)
<<< '.tag=0' (6)
<<< END-OF-SENTENCE

>>> '!done' (5)
>>> 'ret=cb21408d7123ebfc96bec24effe3409f' (36)
>>> '.tag=0' (6)
>>> END-OF SENTENCE

<<< '/login' (6)
<<< '=name=admin' (11)
<<< '=response=0ce20d1ed4bd3ef821dc203a1ff2698461' (44)
<<< '.tag=1' (6)
<<< END-OF-SENTENCE

>>> '!done' (5)
>>> '.tag=1' (6)
>>> END-OF SENTENCE
```

```
<<< '/interface/monitor-traffic' (26)
<<< '=interface=ether1' (17)
<<< '=.proplist=rx-bits-per-second,tx-bits-per-second' (48)
<<< '.tag=2' (6)
<<< END-OF-SENTENCE

>>> '!re' (3)
>>> 'rx-bits-per-second=5744844' (26)
>>> 'tx-bits-per-second=61787' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

{"!re"=>nil, "rx-bits-per-second"=>"5744844", "tx-bits-per-second"=>"61787", ".tag"=>"2"}
>>> '!re' (3)
>>> 'rx-bits-per-second=4310298' (26)
>>> 'tx-bits-per-second=44242' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

{"!re"=>nil, "rx-bits-per-second"=>"4310298", "tx-bits-per-second"=>"44242", ".tag"=>"2"}
>>> '!re' (3)
>>> 'rx-bits-per-second=5442059' (26)
>>> 'tx-bits-per-second=63398' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

{"!re"=>nil, "rx-bits-per-second"=>"5442059", "tx-bits-per-second"=>"63398", ".tag"=>"2"}
>>> '!re' (3)
>>> 'rx-bits-per-second=5711572' (26)
>>> 'tx-bits-per-second=63509' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

{"!re"=>nil, "rx-bits-per-second"=>"5711572", "tx-bits-per-second"=>"63509", ".tag"=>"2"}
>>> '!re' (3)
>>> 'rx-bits-per-second=5711572' (26)
>>> 'tx-bits-per-second=63509' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

{"!re"=>nil, "rx-bits-per-second"=>"5711572", "tx-bits-per-second"=>"63509", ".tag"=>"2"}
>>> '!re' (3)
>>> 'rx-bits-per-second=3712135' (26)
>>> 'tx-bits-per-second=28404' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE
```

```
{"!re"=>nil, "rx-bits-per-second"=>"3712135", "tx-bits-per-second"=>"28404", ".tag"=>"2"}
>>> '!re' (3)
>>> 'rx-bits-per-second=4822099' (26)
>>> 'tx-bits-per-second=39160' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

{"!re"=>nil, "rx-bits-per-second"=>"4822099", "tx-bits-per-second"=>"39160", ".tag"=>"2"}
>>> '!re' (3)
>>> 'rx-bits-per-second=4536317' (26)
>>> 'tx-bits-per-second=52562' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

{"!re"=>nil, "rx-bits-per-second"=>"4536317", "tx-bits-per-second"=>"52562", ".tag"=>"2"}
>>> '!re' (3)
>>> 'rx-bits-per-second=3976832' (26)
>>> 'tx-bits-per-second=45331' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

{"!re"=>nil, "rx-bits-per-second"=>"3976832", "tx-bits-per-second"=>"45331", ".tag"=>"2"}
>>> '!re' (3)
>>> 'rx-bits-per-second=4124776' (26)
>>> 'tx-bits-per-second=80547' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

{"!re"=>nil, "rx-bits-per-second"=>"4124776", "tx-bits-per-second"=>"80547", ".tag"=>"2"}
<<< '/cancel' (7)
<<< '=tag=2' (6)
<<< '.tag=3' (6)
<<< END-OF-SENTENCE

>>> '!trap' (5)
>>> 'category=2' (10)
>>> 'message=interrupted' (19)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!done' (5)
>>> '.tag=3' (6)
>>> END-OF SENTENCE

>>> '!done' (5)
>>> '.tag=2' (6)
```

```
>>> END-OF SENTENCE
```

Updating DNS settings on multiple devices

Imagine I have a list of RouterOS devices that all need primary and secondary DNS settings updated. Here's an example Ruby script to do this:

```
#!/usr/bin/env ruby

require 'rubygems'
require 'mtik'

## List of devices (hostnames/IPs) to contact:
devlist = [
  '10.0.0.4',
  '10.0.0.5',
  '10.0.0.22',
  '10.1.44.22',
  '10.1.44.79'
]

## Example assumes all devices use the same API user/pass:
USERNAME = 'admin'
PASSWORD = 'password'

## Set DNS to these IPs (if these were the name servers in question):
PRIMARYDNS   = '10.20.30.2'
SECONDARYDNS = '192.168.44.2'

## Set to 1 to do each device serially, or greater to fork parallel processes
MAXFORK = 1

children = 0
devlist.each do |host|
  Kernel.fork do
    puts "#{host}: Connecting..."
    mt = nil
    begin
      mt = MTik::Connection.new(
        :host=>host,
        :user=>USERNAME,
        :pass=>PASSWORD
      )
    rescue Errno::ETIMEDOUT, Errno::ENETUNREACH, Errno::EHOSTUNREACH => e
      puts "#{host}: Error connecting: #{e}"
      exit
    end

    ## The MTik::Connection#get_reply() method executes a command, then waits
```

```
    ## for it to complete (either with a '!done' or '!trap' response) before
    ## executing the callback code block.  The call will block (execution of
    ## this script halts) and wait for the command to finish.  Don't use this
    ## method if you need to handle simultaneous commands to a single device
    ## over a single API connection.  Use an asynchronous calls send_request()
    ## and wait_for_reply().
    mt.get_reply(
      '/ip/dns/set',
      "=primary-dns=#{PRIMARYDNS}",
      "=secondary-dns=#{SECONDARYDNS}"
    ) do |request, sentence|
      trap = request.reply.find_sentence('!trap')
      if trap.nil?
        puts "#{host}: Update command was sent."
      else
        puts "#{host}: An error occurred while setting DNS servers: #{trap['message']}"
      end
    end

    ## Now let's double-check the settings:
    mt.get_reply('/ip/dns/getall') do |request, sentence|
      trap = request.reply.find_sentence('!trap')
      if trap.nil?
        re = request.reply.find_sentence('!re')
        unless re.nil?
          ## Check DNS settings:
          if re['primary-dns'] == PRIMARYDNS && re['secondary-dns'] == SECONDARYDNS
            puts "#{host}: Successfully updated DNS servers."
          else
            puts "#{host}: WARNING: DNS servers DO NOT MATCH: primary-dns=" +
                 "'#{re['primary-dns']}', secondary-dns='#{re['secondary-dns']}'"
          end
        else
          puts "#{host}: WARNING: '/ip/dns/getall' command did work to retrieve DNS settings!"
        end
      else
        puts "#{host}: An error occurred while setting DNS servers: #{trap['message']}"
      end
    end
    mt.close
  end
  children += 1
  while children >= MAXFORK
    Process.wait
    children -= 1
  end
end
```

```
while children > 1
  Process.wait
  children -= 1
end
```

Output might look a bit like:

```
user@host:~/$ ./dnsupdate.rb
10.0.0.4: Connecting...
10.0.0.4: Update command was sent.
10.0.0.4: Successfully updated DNS servers.
10.0.0.5: Connecting...
10.0.0.5: Update command was sent.
10.0.0.5: Successfully updated DNS servers.

... MORE OUTPUT ...

10.1.44.79: Successfully updated DNS servers.
user@host:~/$
```

The benefit of using Mikrotik's API is you can whip up a script to do something, then feed it a bunch of device IPs, login user IDs and passwords from a database, then execute desired commands on ALL of the devices. Check settings, change settings, monitor stats, etc.

Changing user group or deleting a user from a device

This shows how one can query a device for a list of configured users, then subsequently use API commands to remove or alter settings for users (if they exist) referencing them by API .id:

```
## Retrieve a list of all users on a RouterOS device (with associated IDs):
users = {}
mt.get_reply_each('/user/getall') do |r, s|
 if s.key?('!re') && s.key?('name')
   users[s['name']] = {
     :name     => s['name'],
     :group    => s['group'],
     :address  => s['address'],
     :comment  => s['comment'],
     :disabled => s['disabled'] == true,
     :id       => s['.id']
   }
 end
end

## Remove a specific named user (if found on the device):
mt.get_reply('/user/remove', "=.id=#{users['foo'][:id]}") if users.key?('foo')

## Make a named user (if found) an read-only user:
mt.get_reply('/user/set', "=.id=#{users['foo'][:id]}", "=group=read")
```

One may wonder why not use `?name=foo` instead of using the ID parameter. The gem author has discovered that API commands that make changes often do not work even though the API does not respond with an error, unless the object to be changed is directly referenced by `.id`. In updating or removing users, this appears to be the case.

Execute a multiple-response command and automatically cancel it to limit the number of replies

```
require 'rubygems'
require 'mtik'

# Be verbose in output
MTik::verbose = true

# Connect to the device:
p connection = MTik::command(
  :host    => '10.0.0.1',
  :user    => 'username',
  :pass    => 'password',
  :command => [
    "/interface/monitor-traffic",
    "=interface=ether0",
    "=.proplist=rx-bits-per-second,tx-bits-per-second"
  ],
  :limit => 10  ## Auto-cancel after 10 replies
)
```

In the above code, the `/interface/monitor-traffic command` is executed using the blocking (non-event-style) MTik::command() library method. But because the `monitor-traffic` command normally will keep sending replies (one per second) forever, the `:limit => 10` parameter was passed. That makes the library count replies and automatically issue a `/cancel` API command after the specified number of replies have been received. That way the blocking-style `MTik::command()` method can be safely used without the program hanging forever.

Below is what this example might output. **REMEMBER** that `MTik::verbose = true` so most of the output is due to that, and only the final line is the actual final data returned by the `MTik::command()` call:

```
<<< '/login' (6)
<<< '.tag=0' (6)
<<< END-OF-SENTENCE

>>> '!done' (5)
>>> 'ret=2830ce30c78f9123d31544654d28a6e0' (36)
>>> '.tag=0' (6)
>>> END-OF SENTENCE

<<< '/login' (6)
<<< '=name=username' (9)
<<< '=response=008a9eba141f9e0f8d2337ab84366298cb' (44)
<<< '.tag=1' (6)
<<< END-OF-SENTENCE
```

```
>>> '!done' (5)
>>> '.tag=1' (6)
>>> END-OF SENTENCE

<<< '/interface/monitor-traffic' (26)
<<< '=interface=ether0' (21)
<<< '=.proplist=rx-bits-per-second,tx-bits-per-second' (48)
<<< '.tag=2' (6)
<<< END-OF-SENTENCE

>>> '!re' (3)
>>> 'rx-bits-per-second=326944' (25)
>>> 'tx-bits-per-second=1175812' (26)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> 'rx-bits-per-second=84251' (24)
>>> 'tx-bits-per-second=444610' (25)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> 'rx-bits-per-second=111604' (25)
>>> 'tx-bits-per-second=21782' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> 'rx-bits-per-second=116277' (25)
>>> 'tx-bits-per-second=681' (22)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> 'rx-bits-per-second=116277' (25)
>>> 'tx-bits-per-second=681' (22)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> 'rx-bits-per-second=339747' (25)
>>> 'tx-bits-per-second=14495' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE
```

```
>>> '!re' (3)
>>> 'rx-bits-per-second=106012' (25)
>>> 'tx-bits-per-second=3952' (23)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> 'rx-bits-per-second=118867' (25)
>>> 'tx-bits-per-second=17370' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> 'rx-bits-per-second=130582' (25)
>>> 'tx-bits-per-second=29941' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!re' (3)
>>> 'rx-bits-per-second=130582' (25)
>>> 'tx-bits-per-second=29941' (24)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

<<< '/cancel' (7)
<<< '=tag=2' (6)
<<< '.tag=3' (6)
<<< END-OF-SENTENCE

>>> '!trap' (5)
>>> 'category=2' (10)
>>> 'message=interrupted' (19)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

>>> '!done' (5)
>>> '.tag=3' (6)
>>> END-OF SENTENCE

>>> '!done' (5)
>>> '.tag=2' (6)
>>> END-OF SENTENCE

<<< '/quit' (5)
<<< '.tag=4' (6)
<<< END-OF-SENTENCE
```

```
>>> '!fatal' (6)
>>> 'session terminated on request' (29)
>>> END-OF SENTENCE

[[{"!re"=>nil, "rx-bits-per-second"=>"326944", "tx-bits-per-second"=>"1175812", ".tag"=>"2"},
 {"!re"=>nil, "rx-bits-per-second"=>"84251", "tx-bits-per-second"=>"444610", ".tag"=>"2"},
 {"!re"=>nil, "rx-bits-per-second"=>"111604", "tx-bits-per-second"=>"21782", ".tag"=>"2"},
 {"!re"=>nil, "rx-bits-per-second"=>"116277", "tx-bits-per-second"=>"681", ".tag"=>"2"},
 {"!re"=>nil, "rx-bits-per-second"=>"116277", "tx-bits-per-second"=>"681", ".tag"=>"2"},
 {"!re"=>nil, "rx-bits-per-second"=>"339747", "tx-bits-per-second"=>"14495", ".tag"=>"2"},
 {"!re"=>nil, "rx-bits-per-second"=>"106012", "tx-bits-per-second"=>"3952", ".tag"=>"2"},
 {"!re"=>nil, "rx-bits-per-second"=>"118867", "tx-bits-per-second"=>"17370", ".tag"=>"2"},
 {"!re"=>nil, "rx-bits-per-second"=>"130582", "tx-bits-per-second"=>"29941", ".tag"=>"2"},
 {"!re"=>nil, "rx-bits-per-second"=>"130582", "tx-bits-per-second"=>"29941", ".tag"=>"2"},
 {"!trap"=>nil, "category"=>"2", "message"=>"interrupted", ".tag"=>"2"},
 {"!done"=>nil, ".tag"=>"2"}]]
```

Notes

- This has only been testing using Ruby 1.9.2 and Ruby 1.8.7 on several FreeBSD hosts, though it should work identically on other Ruby installations.
- Encoding/decoding longer words has NOT be thoroughly tested.
- Connection timeouts and auto-reconnections are NOT implemented.
- The above examples are single-threaded, but it is probably be safe to use within a multi-threaded Ruby application (untested).
- The gem uses an event driven callback style to send commands and receive responses. Multiple simultaneous commands may be executing over a single TCP API connection. If one fully utilizes the event-loop style of programming, one could have a single-threaded single process simultaneously executing many commands over many separate TCP API connections to different devices. To do so, one has to be careful to implement a main event loop to avoid blocking.

See the author's web site in the external links below for a link to the CHANGELOG.

References

[1] http://www.aarongifford.com/computers/mtik/latest/pkg/mtik-4.0.0.gem
[2] http://www.aarongifford.com/computers/mtik/latest/doc/
[3] http://www.aarongifford.com/computers/mtik/latest/bin/tikcli
[4] http://www.aarongifford.com/computers/mtik/latest/bin/tikcommand
[5] http://www.aarongifford.com/computers/mtik/latest/bin/tikfetch
[6] http://www.aarongifford.com/computers/mtik/latest/examples/tikjson.rb

Librouteros

librouteros is a free and open-source library written in C which abstracts the API provided by *RouterOS*. It was initially written in 2009 by Florian Forster and released under the *GNU General Public License* (GPL).

Features and design goals are:

- Ease of use: The library makes heavy use of callback functions which simplifies memory management.
- Strict ISO C99 and POSIX.1-2001 conformance.
- Thread and reentrant-safety.
- Abstraction from underlying network protocol.
- Well documented using manual pages.

Compiling

librouteros uses the *autotools* and *libtool* and can therefore be compiled and installed with the usual set of commands:

```
~ $ tar jxf librouteros-x.y.z.tar.bz2
~ $ cd  librouteros-x.y.z
librouteros-x.y.z $ ./configure
librouteros-x.y.z $ make
librouteros-x.y.z $ make install
```

Dependencies

librouteros uses the *gcrypt* library from the *GnuPG* project to calculate the MD5-hash required for authenticating.

Example program

The following (untested!) example program demonstrates how to get a list of interfaces from a device running *RouterOS* and print this list to standard output. It is using the high-level function `ros_interface` which provides a list of interfaces on the router. For a more thorough example please tend to the *ros* command line utility which is included with the *librouteros* source code distribution. It demonstrates how to handle generic queries and registration-table entries, too.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>

#include "routeros_api.h"

/* Print the list of interfaces. */
static void print_interface (const ros_interface_t *if)
{
  if (if == NULL)
    return;

  printf ("%s (%s, %s, %s, %s)\n", if->name, if->type,
      if->dynamic ? "dynamic" : "static",
```

```
      if->running ? "running" : "stopped",
      if->enabled ? "enabled" : "disabled");

  print_interface (if->next);
}

/* Callback function that is called by "ros_interface" */
static int handle_interfaces (ros_connection_t *c, const ros_interface_t *if, void *user_data)
{
  print_interface (if);
  return (0);
}

int main (int argc, char **argv)
{
  ros_connection_t *c;

  /* Connect to the router */
  c = ros_connect ("my-router.example.com", ROUTEROS_API_PORT,
      "api_user", "secret");
  if (c == NULL)
  {
    fprintf (stderr, "ros_connect failed: %s\n", strerror (errno));
    exit (EXIT_FAILURE);
  }

  /* Query a list of interfaces and call "handle_interfaces". */
  ros_interface (c, handle_interfaces, /* user data = */ NULL);

  /* Disconnect from the router. */
  ros_disconnect (c);
  exit (EXIT_SUCCESS);
}
```

External links

- librouteros' homepage [1]
- *librouteros(3)* manual page [2]
- ros.c [3], a thorough example program.
- Freshmeat entry [4]
- libgcrypt in the Free Software Directory [5]

References

[1] http://verplant.org/librouteros/
[2] http://verplant.org/librouteros/manpages/librouteros.3.html
[3] http://git.verplant.org/?p=routeros-api.git;a=blob;f=src/ros.c;hb=HEAD
[4] http://freshmeat.net/projects/librouteros
[5] http://directory.fsf.org/project/libgcrypt/

API in C

This is an implementation of the RouterOS API written in C. This implementation relies on the MD5 digest calculation functions written by Aladdin Enterprises ([1]). An endian test (big/little endian) is also used courtesy GRASS Development Team ([2]). All functions/libraries used from other sources are available under open licenses such as GNU Public License.

Pre-requisite MD5 calculation function header file (md5.h)

```
/*
  Copyright (C) 1999, 2002 Aladdin Enterprises.  All rights reserved.

  This software is provided 'as-is', without any express or implied
  warranty.  In no event will the authors be held liable for any damages
  arising from the use of this software.

  Permission is granted to anyone to use this software for any purpose,
  including commercial applications, and to alter it and redistribute it
  freely, subject to the following restrictions:

  1. The origin of this software must not be misrepresented; you must not
     claim that you wrote the original software. If you use this software
     in a product, an acknowledgment in the product documentation would be
     appreciated but is not required.
  2. Altered source versions must be plainly marked as such, and must not be
     misrepresented as being the original software.
  3. This notice may not be removed or altered from any source distribution.

  L. Peter Deutsch
  ghost@aladdin.com

 */
/* $Id: md5.h,v 1.4 2002/04/13 19:20:28 lpd Exp $ */
/*
  Independent implementation of MD5 (RFC 1321).

  This code implements the MD5 Algorithm defined in RFC 1321, whose
  text is available at
      http://www.ietf.org/rfc/rfc1321.txt
  The code is derived from the text of the RFC, including the test suite
```

```
  (section A.5) but excluding the rest of Appendix A.  It does not include
  any code or documentation that is identified in the RFC as being
  copyrighted.

  The original and principal author of md5.h is L. Peter Deutsch
  <ghost@aladdin.com>.  Other authors are noted in the change history
  that follows (in reverse chronological order):

  2002-04-13 lpd Removed support for non-ANSI compilers; removed
      references to Ghostscript; clarified derivation from RFC 1321;
      now handles byte order either statically or dynamically.
  1999-11-04 lpd Edited comments slightly for automatic TOC extraction.
  1999-10-18 lpd Fixed typo in header comment (ansi2knr rather than md5);
      added conditionalization for C++ compilation from Martin
      Purschke <purschke@bnl.gov>.
  1999-05-03 lpd Original version.
 */

#ifndef md5_INCLUDED
#  define md5_INCLUDED

/*
 * This package supports both compile-time and run-time determination of CPU
 * byte order.  If ARCH_IS_BIG_ENDIAN is defined as 0, the code will be
 * compiled to run only on little-endian CPUs; if ARCH_IS_BIG_ENDIAN is
 * defined as non-zero, the code will be compiled to run only on big-endian
 * CPUs; if ARCH_IS_BIG_ENDIAN is not defined, the code will be compiled to
 * run on either big- or little-endian CPUs, but will run slightly less
 * efficiently on either one than if ARCH_IS_BIG_ENDIAN is defined.
 */

typedef unsigned char md5_byte_t; /* 8-bit byte */
typedef unsigned int md5_word_t; /* 32-bit word */

/* Define the state of the MD5 Algorithm. */
typedef struct md5_state_s {
    md5_word_t count[2];      /* message length in bits, lsw first */
    md5_word_t abcd[4];           /* digest buffer */
    md5_byte_t buf[64];           /* accumulate block */
} md5_state_t;

#ifdef __cplusplus
extern "C"
{
#endif

/* Initialize the algorithm. */
```

```
void md5_init(md5_state_t *pms);

/* Append a string to the message. */
void md5_append(md5_state_t *pms, const md5_byte_t *data, int nbytes);

/* Finish the message and return the digest. */
void md5_finish(md5_state_t *pms, md5_byte_t digest[16]);

#ifdef __cplusplus
}  /* end extern "C" */
#endif

#endif /* md5_INCLUDED */
```

Pre-requisite MD5 calculation function source file (md5.c)

```
/*
  Copyright (C) 1999, 2000, 2002 Aladdin Enterprises.  All rights reserved.

  This software is provided 'as-is', without any express or implied
  warranty.  In no event will the authors be held liable for any damages
  arising from the use of this software.

  Permission is granted to anyone to use this software for any purpose,
  including commercial applications, and to alter it and redistribute it
  freely, subject to the following restrictions:

  1. The origin of this software must not be misrepresented; you must not
     claim that you wrote the original software. If you use this software
     in a product, an acknowledgment in the product documentation would be
     appreciated but is not required.
  2. Altered source versions must be plainly marked as such, and must not be
     misrepresented as being the original software.
  3. This notice may not be removed or altered from any source distribution.

  L. Peter Deutsch
  ghost@aladdin.com

 */
/* $Id: md5.c,v 1.6 2002/04/13 19:20:28 lpd Exp $ */
/*
  Independent implementation of MD5 (RFC 1321).

  This code implements the MD5 Algorithm defined in RFC 1321, whose
  text is available at
      http://www.ietf.org/rfc/rfc1321.txt
  The code is derived from the text of the RFC, including the test suite
  (section A.5) but excluding the rest of Appendix A.  It does not include
```

```
  any code or documentation that is identified in the RFC as being
  copyrighted.

  The original and principal author of md5.c is L. Peter Deutsch
  <ghost@aladdin.com>.  Other authors are noted in the change history
  that follows (in reverse chronological order):

  2002-04-13 lpd Clarified derivation from RFC 1321; now handles byte order
      either statically or dynamically; added missing #include <string.h>
      in library.
  2002-03-11 lpd Corrected argument list for main(), and added int return
      type, in test program and T value program.
  2002-02-21 lpd Added missing #include <stdio.h> in test program.
  2000-07-03 lpd Patched to eliminate warnings about "constant is
      unsigned in ANSI C, signed in traditional"; made test program
      self-checking.
  1999-11-04 lpd Edited comments slightly for automatic TOC extraction.
  1999-10-18 lpd Fixed typo in header comment (ansi2knr rather than md5).
  1999-05-03 lpd Original version.
 */

#include "md5.h"
#include <string.h>

#undef BYTE_ORDER      /* 1 = big-endian, -1 = little-endian, 0 = unknown */
#ifdef ARCH_IS_BIG_ENDIAN
#  define BYTE_ORDER (ARCH_IS_BIG_ENDIAN ? 1 : -1)
#else
#  define BYTE_ORDER 0
#endif

#define T_MASK ((md5_word_t)~0)
#define T1 /* 0xd76aa478 */ (T_MASK ^ 0x28955b87)
#define T2 /* 0xe8c7b756 */ (T_MASK ^ 0x173848a9)
#define T3    0x242070db
#define T4 /* 0xc1bdceee */ (T_MASK ^ 0x3e423111)
#define T5 /* 0xf57c0faf */ (T_MASK ^ 0x0a83f050)
#define T6    0x4787c62a
#define T7 /* 0xa8304613 */ (T_MASK ^ 0x57cfb9ec)
#define T8 /* 0xfd469501 */ (T_MASK ^ 0x02b96afe)
#define T9    0x698098d8
#define T10 /* 0x8b44f7af */ (T_MASK ^ 0x74bb0850)
#define T11 /* 0xffff5bb1 */ (T_MASK ^ 0x0000a44e)
#define T12 /* 0x895cd7be */ (T_MASK ^ 0x76a32841)
#define T13    0x6b901122
#define T14 /* 0xfd987193 */ (T_MASK ^ 0x02678e6c)
#define T15 /* 0xa679438e */ (T_MASK ^ 0x5986bc71)
```

```
#define T16    0x49b40821
#define T17 /* 0xf61e2562 */ (T_MASK ^ 0x09e1da9d)
#define T18 /* 0xc040b340 */ (T_MASK ^ 0x3fbf4cbf)
#define T19    0x265e5a51
#define T20 /* 0xe9b6c7aa */ (T_MASK ^ 0x16493855)
#define T21 /* 0xd62f105d */ (T_MASK ^ 0x29d0efa2)
#define T22    0x02441453
#define T23 /* 0xd8a1e681 */ (T_MASK ^ 0x275e197e)
#define T24 /* 0xe7d3fbc8 */ (T_MASK ^ 0x182c0437)
#define T25    0x21e1cde6
#define T26 /* 0xc33707d6 */ (T_MASK ^ 0x3cc8f829)
#define T27 /* 0xf4d50d87 */ (T_MASK ^ 0x0b2af278)
#define T28    0x455a14ed
#define T29 /* 0xa9e3e905 */ (T_MASK ^ 0x561c16fa)
#define T30 /* 0xfcefa3f8 */ (T_MASK ^ 0x03105c07)
#define T31    0x676f02d9
#define T32 /* 0x8d2a4c8a */ (T_MASK ^ 0x72d5b375)
#define T33 /* 0xfffa3942 */ (T_MASK ^ 0x0005c6bd)
#define T34 /* 0x8771f681 */ (T_MASK ^ 0x788e097e)
#define T35    0x6d9d6122
#define T36 /* 0xfde5380c */ (T_MASK ^ 0x021ac7f3)
#define T37 /* 0xa4beea44 */ (T_MASK ^ 0x5b4115bb)
#define T38    0x4bdecfa9
#define T39 /* 0xf6bb4b60 */ (T_MASK ^ 0x0944b49f)
#define T40 /* 0xbebfbc70 */ (T_MASK ^ 0x4140438f)
#define T41    0x289b7ec6
#define T42 /* 0xeaa127fa */ (T_MASK ^ 0x155ed805)
#define T43 /* 0xd4ef3085 */ (T_MASK ^ 0x2b10cf7a)
#define T44    0x04881d05
#define T45 /* 0xd9d4d039 */ (T_MASK ^ 0x262b2fc6)
#define T46 /* 0xe6db99e5 */ (T_MASK ^ 0x1924661a)
#define T47    0x1fa27cf8
#define T48 /* 0xc4ac5665 */ (T_MASK ^ 0x3b53a99a)
#define T49 /* 0xf4292244 */ (T_MASK ^ 0x0bd6ddbb)
#define T50    0x432aff97
#define T51 /* 0xab9423a7 */ (T_MASK ^ 0x546bdc58)
#define T52 /* 0xfc93a039 */ (T_MASK ^ 0x036c5fc6)
#define T53    0x655b59c3
#define T54 /* 0x8f0ccc92 */ (T_MASK ^ 0x70f3336d)
#define T55 /* 0xffeff47d */ (T_MASK ^ 0x00100b82)
#define T56 /* 0x85845dd1 */ (T_MASK ^ 0x7a7ba22e)
#define T57    0x6fa87e4f
#define T58 /* 0xfe2ce6e0 */ (T_MASK ^ 0x01d3191f)
#define T59 /* 0xa3014314 */ (T_MASK ^ 0x5cfebceb)
#define T60    0x4e0811a1
#define T61 /* 0xf7537e82 */ (T_MASK ^ 0x08ac817d)
#define T62 /* 0xbd3af235 */ (T_MASK ^ 0x42c50dca)
```

```
#define T63    0x2ad7d2bb
#define T64 /* 0xeb86d391 */ (T_MASK ^ 0x14792c6e)


static void
md5_process(md5_state_t *pms, const md5_byte_t *data /*[64]*/)
{
    md5_word_t
      a = pms->abcd[0], b = pms->abcd[1],
      c = pms->abcd[2], d = pms->abcd[3];
    md5_word_t t;
#if BYTE_ORDER > 0
    /* Define storage only for big-endian CPUs. */
    md5_word_t X[16];
#else
    /* Define storage for little-endian or both types of CPUs. */
    md5_word_t xbuf[16];
    const md5_word_t *X;
#endif

    {
#if BYTE_ORDER == 0
      /*
       * Determine dynamically whether this is a big-endian or
       * little-endian machine, since we can use a more efficient
       * algorithm on the latter.
       */
      static const int w = 1;

      if (*((const md5_byte_t *)&w)) /* dynamic little-endian */
#endif
#if BYTE_ORDER <= 0           /* little-endian */
      {
          /*
           * On little-endian machines, we can process properly aligned
           * data without copying it.
           */
          if (!((data - (const md5_byte_t *)0) & 3)) {
            /* data are properly aligned */
            X = (const md5_word_t *)data;
          } else {
            /* not aligned */
            memcpy(xbuf, data, 64);
            X = xbuf;
          }
      }
#endif
```

```
#if BYTE_ORDER == 0
      else                  /* dynamic big-endian */
#endif
#if BYTE_ORDER >= 0           /* big-endian */
      {
          /*
           * On big-endian machines, we must arrange the bytes in the
           * right order.
           */
          const md5_byte_t *xp = data;
          int i;

#  if BYTE_ORDER == 0
          X = xbuf;             /* (dynamic only) */
#  else
#    define xbuf X            /* (static only) */
#  endif
          for (i = 0; i < 16; ++i, xp += 4)
            xbuf[i] = xp[0] + (xp[1] << 8) + (xp[2] << 16) + (xp[3] << 24);
      }
#endif
    }

#define ROTATE_LEFT(x, n) (((x) << (n)) | ((x) >> (32 - (n))))

    /* Round 1. */
    /* Let [abcd k s i] denote the operation
       a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s). */
#define F(x, y, z) (((x) & (y)) | (~(x) & (z)))
#define SET(a, b, c, d, k, s, Ti)\
  t = a + F(b,c,d) + X[k] + Ti;\
  a = ROTATE_LEFT(t, s) + b
    /* Do the following 16 operations. */
    SET(a, b, c, d,  0,  7,  T1);
    SET(d, a, b, c,  1, 12,  T2);
    SET(c, d, a, b,  2, 17,  T3);
    SET(b, c, d, a,  3, 22,  T4);
    SET(a, b, c, d,  4,  7,  T5);
    SET(d, a, b, c,  5, 12,  T6);
    SET(c, d, a, b,  6, 17,  T7);
    SET(b, c, d, a,  7, 22,  T8);
    SET(a, b, c, d,  8,  7,  T9);
    SET(d, a, b, c,  9, 12, T10);
    SET(c, d, a, b, 10, 17, T11);
    SET(b, c, d, a, 11, 22, T12);
    SET(a, b, c, d, 12,  7, T13);
    SET(d, a, b, c, 13, 12, T14);
```

```
    SET(c, d, a, b, 14, 17, T15);
    SET(b, c, d, a, 15, 22, T16);
#undef SET

     /* Round 2. */
     /* Let [abcd k s i] denote the operation
          a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s). */
#define G(x, y, z) (((x) & (z)) | ((y) & ~(z)))
#define SET(a, b, c, d, k, s, Ti)\
  t = a + G(b,c,d) + X[k] + Ti;\
  a = ROTATE_LEFT(t, s) + b
     /* Do the following 16 operations. */
    SET(a, b, c, d,  1,  5, T17);
    SET(d, a, b, c,  6,  9, T18);
    SET(c, d, a, b, 11, 14, T19);
    SET(b, c, d, a,  0, 20, T20);
    SET(a, b, c, d,  5,  5, T21);
    SET(d, a, b, c, 10,  9, T22);
    SET(c, d, a, b, 15, 14, T23);
    SET(b, c, d, a,  4, 20, T24);
    SET(a, b, c, d,  9,  5, T25);
    SET(d, a, b, c, 14,  9, T26);
    SET(c, d, a, b,  3, 14, T27);
    SET(b, c, d, a,  8, 20, T28);
    SET(a, b, c, d, 13,  5, T29);
    SET(d, a, b, c,  2,  9, T30);
    SET(c, d, a, b,  7, 14, T31);
    SET(b, c, d, a, 12, 20, T32);
#undef SET

     /* Round 3. */
     /* Let [abcd k s t] denote the operation
          a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s). */
#define H(x, y, z) ((x) ^ (y) ^ (z))
#define SET(a, b, c, d, k, s, Ti)\
  t = a + H(b,c,d) + X[k] + Ti;\
  a = ROTATE_LEFT(t, s) + b
     /* Do the following 16 operations. */
    SET(a, b, c, d,  5,  4, T33);
    SET(d, a, b, c,  8, 11, T34);
    SET(c, d, a, b, 11, 16, T35);
    SET(b, c, d, a, 14, 23, T36);
    SET(a, b, c, d,  1,  4, T37);
    SET(d, a, b, c,  4, 11, T38);
    SET(c, d, a, b,  7, 16, T39);
    SET(b, c, d, a, 10, 23, T40);
    SET(a, b, c, d, 13,  4, T41);
```

```
    SET(d, a, b, c,  0, 11, T42);
    SET(c, d, a, b,  3, 16, T43);
    SET(b, c, d, a,  6, 23, T44);
    SET(a, b, c, d,  9,  4, T45);
    SET(d, a, b, c, 12, 11, T46);
    SET(c, d, a, b, 15, 16, T47);
    SET(b, c, d, a,  2, 23, T48);
#undef SET

     /* Round 4. */
     /* Let [abcd k s t] denote the operation
          a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s). */
#define I(x, y, z) ((y) ^ ((x) | ~(z)))
#define SET(a, b, c, d, k, s, Ti)\
  t = a + I(b,c,d) + X[k] + Ti;\
  a = ROTATE_LEFT(t, s) + b
     /* Do the following 16 operations. */
    SET(a, b, c, d,  0,  6, T49);
    SET(d, a, b, c,  7, 10, T50);
    SET(c, d, a, b, 14, 15, T51);
    SET(b, c, d, a,  5, 21, T52);
    SET(a, b, c, d, 12,  6, T53);
    SET(d, a, b, c,  3, 10, T54);
    SET(c, d, a, b, 10, 15, T55);
    SET(b, c, d, a,  1, 21, T56);
    SET(a, b, c, d,  8,  6, T57);
    SET(d, a, b, c, 15, 10, T58);
    SET(c, d, a, b,  6, 15, T59);
    SET(b, c, d, a, 13, 21, T60);
    SET(a, b, c, d,  4,  6, T61);
    SET(d, a, b, c, 11, 10, T62);
    SET(c, d, a, b,  2, 15, T63);
    SET(b, c, d, a,  9, 21, T64);
#undef SET

     /* Then perform the following additions. (That is increment each
        of the four registers by the value it had before this block
        was started.) */
    pms->abcd[0] += a;
    pms->abcd[1] += b;
    pms->abcd[2] += c;
    pms->abcd[3] += d;
}

void
md5_init(md5_state_t *pms)
{
```

```
    pms->count[0] = pms->count[1] = 0;
    pms->abcd[0] = 0x67452301;
    pms->abcd[1] = /*0xefcdab89*/ T_MASK ^ 0x10325476;
    pms->abcd[2] = /*0x98badcfe*/ T_MASK ^ 0x67452301;
    pms->abcd[3] = 0x10325476;
}

void
md5_append(md5_state_t *pms, const md5_byte_t *data, int nbytes)
{
    const md5_byte_t *p = data;
    int left = nbytes;
    int offset = (pms->count[0] >> 3) & 63;
    md5_word_t nbits = (md5_word_t)(nbytes << 3);

    if (nbytes <= 0)
      return;

    /* Update the message length. */
    pms->count[1] += nbytes >> 29;
    pms->count[0] += nbits;
    if (pms->count[0] < nbits)
      pms->count[1]++;

    /* Process an initial partial block. */
    if (offset) {
      int copy = (offset + nbytes > 64 ? 64 - offset : nbytes);

      memcpy(pms->buf + offset, p, copy);
      if (offset + copy < 64)
          return;
      p += copy;
      left -= copy;
      md5_process(pms, pms->buf);
    }

    /* Process full blocks. */
    for (; left >= 64; p += 64, left -= 64)
      md5_process(pms, p);

    /* Process a final partial block. */
    if (left)
      memcpy(pms->buf, p, left);
}

void
md5_finish(md5_state_t *pms, md5_byte_t digest[16])
```

```
{
    static const md5_byte_t pad[64] = {
      0x80, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
    };
    md5_byte_t data[8];
    int i;

    /* Save the length before padding. */
    for (i = 0; i < 8; ++i)
      data[i] = (md5_byte_t)(pms->count[i >> 2] >> ((i & 3) << 3));
    /* Pad to 56 bytes mod 64. */
    md5_append(pms, pad, ((55 - (pms->count[0] >> 3)) & 63) + 1);
    /* Append the length. */
    md5_append(pms, data, 8);
    for (i = 0; i < 16; ++i)
      digest[i] = (md5_byte_t)(pms->abcd[i >> 2] >> ((i & 3) << 3));
}
```

RouterOS API Header file (mikrotik-api.h)

This is the API header file.

Notes:

- DEBUG flag is defined for debugging purposes...generates alot of internal data via printf
- DONE, TRAP and FATAL constants are defined
- Sentence and Block structs are defined.
  - Each word in a sentence is stored as a string. Sentence structs contain individual API words (stored as an array of strings).
  - Block structs represent the full API response...an array of sentences. Blocks are not defined in the Mikrotik API specs, but are a convenient way to represent a full API response in the context of this implementation.

```
#include "md5.h"

#define DEBUG 0

#define DONE 1
#define TRAP 2
#define FATAL 3

struct Sentence {
      char **szSentence;    // array of strings representing individual words
      int iLength;          // length of szSentence (number of array elements)
      int iReturnValue;     // return value of sentence reads from API
};

struct Block {
```

```
      struct Sentence **stSentence;
      int iLength;
};

// endianness variable...global
int iLittleEndian;

// API specific functions
int apiConnect(char *, int);
void apiDisconnect(int);
int login(int, char *, char *);
void writeLen(int, int);
int readLen(int);
void writeWord(int, char *);
char *readWord(int);

// API helper functions to make things a little bit easier
void initializeSentence(struct Sentence *);
void clearSentence(struct Sentence *);
void writeSentence(int, struct Sentence *);
struct Sentence readSentence(int);
void printSentence(struct Sentence *);
void addWordToSentence(struct Sentence *, char *);
void addPartWordToSentence(struct Sentence *, char *);

void initializeBlock(struct Block *);
void clearBlock(struct Block *);
struct Block readBlock(int);
void addSentenceToBlock(struct Block *, struct Sentence *);
void printBlock(struct Block *);

// MD5 helper functions
char *md5DigestToHexString(md5_byte_t *);
char *md5ToBinary(char *);
char hexStringToChar(char *);

// Endian tests
int isLittleEndian(void);
```

RouterOS API Source file (mikrotik-api.c)

The code below is fully commented with notes.

```
/**************************************************************

 * Some definitions

 * Word = piece of API code

 * Sentence = multiple words

 * Block = multiple sentences (usually in response to a sentence request)
```

```
 *


      int fdSock;

      int iLoginResult;

      struct Sentence stSentence;

      struct Block stBlock;


      fdSock = apiConnect("10.0.0.1", 8728);


      // attempt login

      iLoginResult = login(fdSock, "admin", "adminPassword");


      if (!iLoginResult)

      {

            apiDisconnect(fdSock);

            printf("Invalid username or password.\n");

            exit(1);

      }


      // initialize, fill and send sentence to the API

      initializeSentence(&stSentence);

      addWordToSentence(&stSentence, "/interface/getall");

      writeSentence(fdSock, &stSentence);


      // receive and print block from the API

      stBlock = readBlock(fdSock);

      printBlock(&stBlock);


      apiDisconnect(fdSock);


 ********************************************************************/


#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<arpa/inet.h>

#include<unistd.h>

#include<string.h>

#include<stdlib.h>

#include "md5.h"

#include "mikrotik-api.h"




/********************************************************************

 * Connect to API
```

```
 * Returns a socket descriptor
 *****************************************************************/
int apiConnect(char *szIPaddr, int iPort)
{
      int fdSock;
      struct sockaddr_in address;
      int iConnectResult;
      int iLen;

      fdSock = socket(AF_INET, SOCK_STREAM, 0);

      address.sin_family = AF_INET;
      address.sin_addr.s_addr = inet_addr(szIPaddr);
      address.sin_port = htons(iPort);
      iLen = sizeof(address);

      DEBUG ? printf("Connecting to %s\n", szIPaddr) : 0;

      iConnectResult = connect(fdSock, (struct sockaddr *)&address, iLen);

      if(iConnectResult==-1)
      {
            perror ("Connection problem");
            exit(1);
      }
      else
      {
            DEBUG ? printf("Successfully connected to %s\n", szIPaddr) : 0;
      }

      // determine endianness of this machine
      // iLittleEndian will be set to 1 if we are
      // on a little endian machine...otherwise
      // we are assumed to be on a big endian processor
      iLittleEndian = isLittleEndian();

      return fdSock;
}



/*****************************************************************
 * Disconnect from API
 * Close the API socket
 *****************************************************************/
void apiDisconnect(int fdSock)
{
```

```
	DEBUG ? printf("Closing socket\n") : 0;

	close(fdSock);
}



/*****************************************************************
 * Login to the API
 * 1 is returned on successful login
 * 0 is returned on unsuccessful login
 *****************************************************************/
int login(int fdSock, char *username, char *password)
{
	struct Sentence stReadSentence;
	struct Sentence stWriteSentence;
	char *szMD5Challenge;
	char *szMD5ChallengeBinary;
	char *szMD5PasswordToSend;
	char *szLoginUsernameResponseToSend;
	char *szLoginPasswordResponseToSend;
	md5_state_t state;
	md5_byte_t digest[16];
	char cNull[1] = {0};



	writeWord(fdSock, "/login");
	writeWord(fdSock, "");

	stReadSentence = readSentence(fdSock);
	DEBUG ? printSentence (&stReadSentence) : 0;

	if (stReadSentence.iReturnValue != DONE)
	{
		printf("error.\n");
		exit(0);
	}

	// extract md5 string from the challenge sentence
	szMD5Challenge = strtok(stReadSentence.szSentence[1], "=");
	szMD5Challenge = strtok(NULL, "=");

	DEBUG ? printf("MD5 of challenge = %s\n", szMD5Challenge) : 0;

	// convert szMD5Challenge to binary
	szMD5ChallengeBinary = md5ToBinary(szMD5Challenge);
```

```
	// get md5 of the password + challenge concatenation
	md5_init(&state);
	md5_append(&state, cNull, 1);
	md5_append(&state, (const md5_byte_t *)password, strlen(password));
	md5_append(&state, (const md5_byte_t *)szMD5ChallengeBinary, strlen(szMD5ChallengeBinary));
	md5_finish(&state, digest);

	// convert this digest to a string representation of the hex values
	// digest is the binary format of what we want to send
	// szMD5PasswordToSend is the "string" hex format
	szMD5PasswordToSend = md5DigestToHexString(digest);

	clearSentence(&stReadSentence);

	DEBUG ? printf("szPasswordToSend = %s\n", szMD5PasswordToSend) : 0;

	// put together the login sentence
	initializeSentence(&stWriteSentence);

	addWordToSentence(&stWriteSentence, "/login");
	addWordToSentence(&stWriteSentence, "=name=");
	addPartWordToSentence(&stWriteSentence, username);
	addWordToSentence(&stWriteSentence, "=response=00");
	addPartWordToSentence(&stWriteSentence, szMD5PasswordToSend);

	DEBUG ? printSentence(&stWriteSentence) : 0;
	writeSentence(fdSock, &stWriteSentence);



	stReadSentence = readSentence(fdSock);
	DEBUG ? printSentence (&stReadSentence) : 0;

	if (stReadSentence.iReturnValue == DONE)
	{
		clearSentence(&stReadSentence);
		return 1;
	}
	else
	{
		clearSentence(&stReadSentence);
		return 0;
	}
}



/*****************************************************************
```

```
 * Encode message length and write it out to the socket

 *****************************************************************/

void writeLen(int fdSock, int iLen)

{

      char *cEncodedLength;  // encoded length to send to the api socket

      char *cLength;         // exactly what is in memory at &iLen integer


      cLength = calloc(sizeof(int), 1);

      cEncodedLength = calloc(sizeof(int), 1);


      // set cLength address to be same as iLen

      cLength = (char *)&iLen;


      DEBUG ? printf("length of word is %d\n", iLen) : 0;


      // write 1 byte

      if (iLen < 0x80)

      {

            cEncodedLength[0] = (char)iLen;

            write (fdSock, cEncodedLength, 1);

      }


      // write 2 bytes

      else if (iLen < 0x4000)

      {

            DEBUG ? printf("iLen < 0x4000.\n") : 0;


            if (iLittleEndian)

            {

                  cEncodedLength[0] = cLength[1] | 0x80;

                  cEncodedLength[1] = cLength[0];

            }

            else

            {

                  cEncodedLength[0] = cLength[2] | 0x80;

                  cEncodedLength[1] = cLength[3];

            }


            write (fdSock, cEncodedLength, 2);

      }


      // write 3 bytes

       else if (iLen < 0x200000)

      {

            DEBUG ? printf("iLen < 0x200000.\n") : 0;


            if (iLittleEndian)
```

```
            {
                  cEncodedLength[0] = cLength[2] | 0xc0;
                  cEncodedLength[1] = cLength[1];
                  cEncodedLength[2] = cLength[0];
            }
            else
            {
                  cEncodedLength[0] = cLength[1] | 0xc0;
                  cEncodedLength[1] = cLength[2];
                  cEncodedLength[2] = cLength[3];
            }

            write (fdSock, cEncodedLength, 3);
      }

      // write 4 bytes
      // this code SHOULD work, but is untested...
      else if (iLen < 0x10000000)
      {
            DEBUG ? printf("iLen < 0x10000000.\n") : 0;

            if (iLittleEndian)
            {
                  cEncodedLength[0] = cLength[3] | 0xe0;
                  cEncodedLength[1] = cLength[2];
                  cEncodedLength[2] = cLength[1];
                  cEncodedLength[3] = cLength[0];
            }
            else
            {
                  cEncodedLength[0] = cLength[0] | 0xe0;
                  cEncodedLength[1] = cLength[1];
                  cEncodedLength[2] = cLength[2];
                  cEncodedLength[3] = cLength[3];
            }

            write (fdSock, cEncodedLength, 4);
      }
      else  // this should never happen
      {
            printf("length of word is %d\n", iLen);
            printf("word is too long.\n");
            exit(1);
      }
}
```

```
/*********************************************************************
 * Write a word to the socket
 *********************************************************************/
void writeWord(int fdSock, char *szWord)
{
      DEBUG ? printf("Word to write is %s\n", szWord) : 0;
      writeLen(fdSock, strlen(szWord));
      write(fdSock, szWord, strlen(szWord));
}




/*********************************************************************
 * Write a sentence (multiple words) to the socket
 *********************************************************************/
void writeSentence(int fdSock, struct Sentence *stWriteSentence)
{
      int iIndex;

      if (stWriteSentence->iLength == 0)
      {
            return;
      }

      DEBUG ? printf("Writing sentence\n"): 0;
      DEBUG ? printSentence(stWriteSentence) : 0;

      for (iIndex=0; iIndex<stWriteSentence->iLength; iIndex++)
      {
            writeWord(fdSock, stWriteSentence->szSentence[iIndex]);
      }

      writeWord(fdSock, "");
}




/*********************************************************************
 * Read a message length from the socket
 *
 * 80 = 10000000 (2 character encoded length)
 * C0 = 11000000 (3 character encoded length)
 * E0 = 11100000 (4 character encoded length)
 *
 * Message length is returned
 *********************************************************************/
```

```
int readLen(int fdSock)

{

	char cFirstChar; // first character read from socket

	char *cLength;   // length of next message to read...will be cast to int at the end

	int *iLen;       // calculated length of next message (Cast to int)


	cLength = calloc(sizeof(int), 1);


	DEBUG ? printf("start readLen()\n") : 0;


	read(fdSock, &cFirstChar, 1);


	DEBUG ? printf("byte1 = %#x\n", cFirstChar) : 0;


	// read 4 bytes

	// this code SHOULD work, but is untested...

	if ((cFirstChar & 0xE0) == 0xE0)

	{

		DEBUG ? printf("4-byte encoded length\n") : 0;


		if (iLittleEndian)

		{

			cLength[3] = cFirstChar;

			cLength[3] &= 0x1f;        // mask out the 1st 3 bits

			read(fdSock, &cLength[2], 1);

			read(fdSock, &cLength[1], 1);

			read(fdSock, &cLength[0], 1);

		}

		else

		{

			cLength[0] = cFirstChar;

			cLength[0] &= 0x1f;        // mask out the 1st 3 bits

			read(fdSock, &cLength[1], 1);

			read(fdSock, &cLength[2], 1);

			read(fdSock, &cLength[3], 1);

		}


		iLen = (int *)cLength;

	}


	// read 3 bytes

	else if ((cFirstChar & 0xC0) == 0xC0)

	{

		DEBUG ? printf("3-byte encoded length\n") : 0;


		if (iLittleEndian)

		{
```

```
                cLength[2] = cFirstChar;
                cLength[2] &= 0x3f;        // mask out the 1st 2 bits
                read(fdSock, &cLength[1], 1);
                read(fdSock, &cLength[0], 1);
          }
          else
          {
                cLength[1] = cFirstChar;
                cLength[1] &= 0x3f;        // mask out the 1st 2 bits
                read(fdSock, &cLength[2], 1);
                read(fdSock, &cLength[3], 1);
          }

          iLen = (int *)cLength;
    }

    // read 2 bytes
    else if ((cFirstChar & 0x80) == 0x80)
    {
          DEBUG ? printf("2-byte encoded length\n") : 0;

          if (iLittleEndian)
          {
                cLength[1] = cFirstChar;
                cLength[1] &= 0x7f;        // mask out the 1st bit
                read(fdSock, &cLength[0], 1);
          }
          else
          {
                cLength[2] = cFirstChar;
                cLength[2] &= 0x7f;        // mask out the 1st bit
                read(fdSock, &cLength[3], 1);
          }

          iLen = (int *)cLength;
    }

    // assume 1-byte encoded length...same on both LE and BE systems
    else
    {
          DEBUG ? printf("1-byte encoded length\n") : 0;
          iLen = malloc(sizeof(int));
          *iLen = (int)cFirstChar;
    }

    return *iLen;
}
```

```
/*****************************************************************
 * Read a word from the socket
 * The word that was read is returned as a string
 *****************************************************************/
char *readWord(int fdSock)
{
      int iLen = readLen(fdSock);
      int iBytesToRead = 0;
      int iBytesRead = 0;
      char *szWord;
      char *szRetWord;
      char *szTmpWord;

      DEBUG ? printf("readWord iLen=%x\n", iLen) : 0;

      if (iLen > 0)
      {
            // allocate memory for strings
            szRetWord = calloc(sizeof(char), iLen + 1);
            szTmpWord = calloc(sizeof(char), 1024 + 1);

            while (iLen != 0)
            {
                  // determine number of bytes to read this time around
                  // lesser of 1024 or the number of byes left to read
                  // in this word
                  iBytesToRead = iLen > 1024 ? 1024 : iLen;

                  // read iBytesToRead from the socket
                  iBytesRead = read(fdSock, szTmpWord, iBytesToRead);

                  // terminate szTmpWord
                  szTmpWord[iBytesRead] = 0;

                  // concatenate szTmpWord to szRetWord
                  strcat(szRetWord, szTmpWord);

                  // subtract the number of bytes we just read from iLen
                  iLen -= iBytesRead;
            }

            // deallocate szTmpWord
```

```
            free(szTmpWord);


            DEBUG ? printf("word = %s\n", szRetWord) : 0;
            return szRetWord;
      }
      else
      {
            return NULL;
      }
}




/*****************************************************************
 * Read a sentence from the socket
 * A Sentence struct is returned
 *****************************************************************/
struct Sentence readSentence(int fdSock)
{
      struct Sentence stReturnSentence;
      char *szWord;
      int i=0;
      int iReturnLength=0;


      DEBUG ? printf("readSentence\n") : 0;


      initializeSentence(&stReturnSentence);


      while (szWord = readWord(fdSock))
      {
            addWordToSentence(&stReturnSentence, szWord);


            // check to see if we can get a return value from the API
            if (strstr(szWord, "!done") != NULL)
            {
                  DEBUG ? printf("return sentence contains !done\n") : 0;
                  stReturnSentence.iReturnValue = DONE;
            }
            else if (strstr(szWord, "!trap") != NULL)
            {
                  DEBUG ? printf("return sentence contains !trap\n") : 0;
                  stReturnSentence.iReturnValue = TRAP;
            }
            else if (strstr(szWord, "!fatal") != NULL)
            {
                  DEBUG ? printf("return sentence contains !fatal\n") : 0;
                  stReturnSentence.iReturnValue = FATAL;
```

```
            }


      }


      // if any errors, get the next sentence
      if (stReturnSentence.iReturnValue == TRAP || stReturnSentence.iReturnValue == FATAL)
      {
            readSentence(fdSock);
      }


      if (DEBUG)
      {
            for (i=0; i<stReturnSentence.iLength; i++)
            {
                  printf("stReturnSentence.szSentence[%d] = %s\n", i, stReturnSentence.szSentence[i]);
            }
      }


      return stReturnSentence;
}




/*****************************************************************
 * Read sentence block from the socket...keeps reading sentences
 * until it encounters !done, !trap or !fatal from the socket
 *****************************************************************/
struct Block readBlock(int fdSock)
{
      struct Sentence stSentence;
    struct Block stBlock;
      initializeBlock(&stBlock);


      DEBUG ? printf("readBlock\n") : 0;


      do
      {
            stSentence = readSentence(fdSock);
            DEBUG ? printf("readSentence succeeded.\n") : 0;


            addSentenceToBlock(&stBlock, &stSentence);
            DEBUG ? printf("addSentenceToBlock succeeded\n") : 0;


      } while (stSentence.iReturnValue == 0);



      DEBUG ? printf("readBlock completed successfully\n") : 0;
```

```
      return stBlock;
}




/*****************************************************************
 * Initialize a new block
 * Set iLength to 0.
 *****************************************************************/
void initializeBlock(struct Block *stBlock)
{
      DEBUG ? printf("initializeBlock\n") : 0;

      stBlock->iLength = 0;
}



/*****************************************************************
 * Clear an existing block
 * Free all sentences in the Block struct and set iLength to 0.
 *****************************************************************/
void clearBlock(struct Block *stBlock)
{
      DEBUG ? printf("clearBlock\n") : 0;

      free(stBlock->stSentence);
      initializeBlock(stBlock);
}




/*****************************************************************
 * Print a block.
 * Output a Block with printf.
 *****************************************************************/
void printBlock(struct Block *stBlock)
{
      int i;

      DEBUG ? printf("printBlock\n") : 0;
      DEBUG ? printf("block iLength = %d\n", stBlock->iLength) : 0;

      for (i=0; i<stBlock->iLength; i++)
      {
            printSentence(stBlock->stSentence[i]);
      }
```

```
}




/*****************************************************************
 * Add a sentence to a block
 * Allocate memory and add a sentence to a Block.
 *****************************************************************/
void addSentenceToBlock(struct Block *stBlock, struct Sentence *stSentence)
{
      int iNewLength;
      iNewLength = stBlock->iLength + 1;

      DEBUG ? printf("addSentenceToBlock iNewLength=%d\n", iNewLength) : 0;

      // allocate mem for the new Sentence position
      if (stBlock->iLength == 0)
      {
            stBlock->stSentence = malloc(1 * sizeof stBlock->stSentence);
      }
      else
      {
            stBlock->stSentence = realloc(stBlock->stSentence, iNewLength * sizeof stBlock->stSentence + 1);
      }



      // allocate mem for the full sentence struct
      stBlock->stSentence[stBlock->iLength] = malloc(sizeof *stSentence);

      // copy actual sentence struct to the block position
      memcpy(stBlock->stSentence[stBlock->iLength], stSentence, sizeof *stSentence);

      // update iLength
      stBlock->iLength = iNewLength;

      DEBUG ? printf("addSentenceToBlock stBlock->iLength=%d\n", stBlock->iLength) : 0;
}




/*****************************************************************
 * Initialize a new sentence
 *****************************************************************/
void initializeSentence(struct Sentence *stSentence)
{
      DEBUG ? printf("initializeSentence\n") : 0;
```

```
	stSentence->iLength = 0;
	stSentence->iReturnValue = 0;
}




/*****************************************************************
 * Clear an existing sentence
 *****************************************************************/
void clearSentence(struct Sentence *stSentence)
{
	DEBUG ? printf("initializeSentence\n") : 0;

	free(stSentence->szSentence);
	initializeSentence(stSentence);
}




/*****************************************************************
 * Add a word to a sentence struct
 *****************************************************************/
void addWordToSentence(struct Sentence *stSentence, char *szWordToAdd)
{
	int iNewLength;
	iNewLength = stSentence->iLength + 1;

	// allocate mem for the new word position
	if (stSentence->iLength == 0)
	{
		stSentence->szSentence = malloc(1 * sizeof stSentence->szSentence);
	}
	else
	{
		stSentence->szSentence = realloc(stSentence->szSentence, iNewLength * sizeof stSentence->szSentence + 1);
	}



	// allocate mem for the full word string
	stSentence->szSentence[stSentence->iLength] = malloc(strlen(szWordToAdd) + 1);

	// copy word string to the sentence
	strcpy(stSentence->szSentence[stSentence->iLength], szWordToAdd);

	// update iLength
	stSentence->iLength = iNewLength;
}
```

```
/*********************************************************************
 * Add a partial word to a sentence struct...useful for concatenation
 *********************************************************************/
void addPartWordToSentence(struct Sentence *stSentence, char *szWordToAdd)
{
      int iIndex;
      iIndex = stSentence->iLength - 1;

      // reallocate memory for the new partial word
      stSentence->szSentence[iIndex] = realloc(stSentence->szSentence[iIndex], strlen(stSentence->szSentence[iIndex]) + strlen(szWordToAdd) + 1);

      // concatenate the partial word to the existing sentence
      strcat (stSentence->szSentence[iIndex], szWordToAdd);
}



/*********************************************************************
 * Print a Sentence struct
 *********************************************************************/
void printSentence(struct Sentence *stSentence)
{
      int i;

      DEBUG ? printf("Sentence iLength = %d\n", stSentence->iLength) : 0;
      DEBUG ? printf("Sentence iReturnValue = %d\n", stSentence->iReturnValue) : 0;

      printf("Sentence iLength = %d\n", stSentence->iLength);
      printf("Sentence iReturnValue = %d\n", stSentence->iReturnValue);

      for (i=0; i<stSentence->iLength; i++)
      {
            printf(">>> %s\n", stSentence->szSentence[i]);
      }

      printf("\n");
}



/*********************************************************************
 * MD5 helper function to convert an md5 hex char representation to
 * binary representation.
```

```
 ****************************************************************/

char *md5ToBinary(char *szHex)

{

      int di;

      char cBinWork[3];

      char *szReturn;


      // allocate 16 + 1 bytes for our return string

      szReturn = malloc((16 + 1) * sizeof *szReturn);


      // 32 bytes in szHex?

      if (strlen(szHex) != 32)

      {

            return NULL;

      }


      for (di=0; di<32; di+=2)

      {

            cBinWork[0] = szHex[di];

            cBinWork[1] = szHex[di + 1];

            cBinWork[2] = 0;


            DEBUG ? printf("cBinWork = %s\n", cBinWork) : 0;


            szReturn[di/2] = hexStringToChar(cBinWork);

      }


      return szReturn;

}




/****************************************************************

 * MD5 helper function to calculate and return hex representation

 * of an MD5 digest stored in binary.

 ****************************************************************/

char *md5DigestToHexString(md5_byte_t *binaryDigest)

{

      int di;

      char *szReturn;


      // allocate 32 + 1 bytes for our return string

      szReturn = malloc((32 + 1) * sizeof *szReturn);



      for (di = 0; di < 16; ++di)

      {
```

```
            sprintf(szReturn + di * 2, "%02x", binaryDigest[di]);
      }


      return szReturn;
}




/*****************************************************************
 * Quick and dirty function to convert hex string to char...
 * the toConvert string MUST BE 2 characters + null terminated.
 *****************************************************************/
char hexStringToChar(char *cToConvert)
{
      char cConverted;
      unsigned int iAccumulated=0;
      char cString0[2] = {cToConvert[0], 0};
      char cString1[2] = {cToConvert[1], 0};


      // look @ first char in the 16^1 place
      if (cToConvert[0] == 'f' || cToConvert[0] == 'F')
      {
            iAccumulated += 16*15;
      }
      else if (cToConvert[0] == 'e' || cToConvert[0] == 'E')
      {
            iAccumulated += 16*14;
      }
      else if (cToConvert[0] == 'd' || cToConvert[0] == 'D')
      {
            iAccumulated += 16*13;
      }
      else if (cToConvert[0] == 'c' || cToConvert[0] == 'C')
      {
            iAccumulated += 16*12;
      }
      else if (cToConvert[0] == 'b' || cToConvert[0] == 'B')
      {
            iAccumulated += 16*11;
      }
      else if (cToConvert[0] == 'a' || cToConvert[0] == 'A')
      {
            iAccumulated += 16*10;
      }
      else
      {
            iAccumulated += 16 * atoi(cString0);
```

```
	}



	// now look @ the second car in the 16^0 place
	if (cToConvert[1] == 'f' || cToConvert[1] == 'F')
	{
		iAccumulated += 15;
	}
	else if (cToConvert[1] == 'e' || cToConvert[1] == 'E')
	{
		iAccumulated += 14;
	}
	else if (cToConvert[1] == 'd' || cToConvert[1] == 'D')
	{
		iAccumulated += 13;
	}
	else if (cToConvert[1] == 'c' || cToConvert[1] == 'C')
	{
		iAccumulated += 12;
	}
	else if (cToConvert[1] == 'b' || cToConvert[1] == 'B')
	{
		iAccumulated += 11;
	}
	else if (cToConvert[1] == 'a' || cToConvert[1] == 'A')
	{
		iAccumulated += 10;
	}
	else
	{
		iAccumulated += atoi(cString1);
	}


	DEBUG ? printf("%d\n", iAccumulated) : 0;
	return (char)iAccumulated;
}







/*****************************************************************
 * Test whether or not this system is little endian at RUNTIME
 * Courtesy: http://download.osgeo.org/grass/grass6_progman/endian_8c_source.html
 *****************************************************************/
int isLittleEndian(void)
{
	union
```

```
    {
        int testWord;
        char testByte[sizeof(int)];
    } endianTest;

    endianTest.testWord = 1;

    if (endianTest.testByte[0] == 1)
    return 1;              /* true: little endian */

    return 0;                  /* false: big endian */
}
```

Sample Client (mikrotik-tty.c)

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<unistd.h>
#include<string.h>
#include<stdlib.h>
#include "mikrotik-api.h"

/**********************************************************************
 * Print program usage
 **********************************************************************/
void usage()
{
      printf("Usage: mikrotik-tty [-u<username>] [-p<password>] [-P<portNum>] [--quiet] <ip_address>\n\n");
      printf("-u<username> the username to login as.  Default is admin\n");
      printf("-p<password> the password to use for login.  Default is empty string\n");
      printf("-P<port> TCP port to use for API connection.  Default is 8728.\n");
      printf("--quiet Suppress all non-API output.  Default is interactive mode.\n");
      printf("<ip_address> IP address to connect to.  REQUIRED\n\n");
}




/**********************************************************************
 * main
 **********************************************************************/
int main(int argc, char *argv[])
{
      // declare variables
      int fdSock;
```

```
char *szIPaddr;
char *szPort = "8728"; // default port string
int iPort;             // default port int
char *szUsername = "admin";  // default username
char *szPassword = "";       // default password
int iInteractiveMode = 1;    // interactive mode...if set to 0, will supress all non-API output

int iLoginResult;
int iIndex;
char cWordInput[256];        // limit user word input to 256 chars
char *szNewline;             // used for word input from the user
struct Sentence stSentence;
struct Block stBlock;

// check number of args.  if not correct, call usage and exit
if (argc < 2)
{
      usage();
      exit(0);
}

// parse command line parameters
else
{
      for (iIndex=0; iIndex<argc; iIndex++)
      {
            if (strstr(argv[iIndex], "-u"))
            {
                  szUsername = &argv[iIndex][2];
            }
            else if (strstr(argv[iIndex], "-p"))
            {
                  szPassword = &argv[iIndex][2];
            }
            else if (strstr(argv[iIndex], "-P"))
            {
                  szPort = &argv[iIndex][2];
            }
            else if (strstr(argv[iIndex], "--quiet"))
            {
                  iInteractiveMode = 0;
            }
      }

      // assume the last parameter is the IP address
      szIPaddr = argv[argc-1];
```

```
		// convert port string to an int
		iPort = atoi(szPort);
	}

	iInteractiveMode ? printf("Connecting to API: %s:%d\n", szIPaddr, iPort) : 0;
	fdSock = apiConnect(szIPaddr, iPort);

	iLoginResult = login(fdSock, szUsername, szPassword);

	if (!iLoginResult)
	{
		apiDisconnect(fdSock);
		iInteractiveMode ? printf("Invalid username or password.\n") : 0;
		exit(1);
	}

	// initialize first sentence
	initializeSentence(&stSentence);

	// main loop
	while (1)
	{
		// get input from stdin
		iInteractiveMode ? fputs("<<< ", stdout): 0;
		iInteractiveMode ? fflush(stdout): 0;

		if (fgets(cWordInput, sizeof cWordInput, stdin) != NULL)
		{
			szNewline = strchr(cWordInput, '\n');

			if (szNewline != NULL)
			{
				*szNewline = '\0';
			}
		}

		// check to see if we want to quit
		if (strcmp(cWordInput, "quit") == 0)
		{
			break;
		}

		// check for end of sentence (\n)
		else if (strcmp(cWordInput, "") == 0)
		{
			// write sentence to the API
			if (stSentence.iLength > 0)
```

```
                {
                        writeSentence(fdSock, &stSentence);

                        // receive and print response block from the API
                        stBlock = readBlock(fdSock);
                        printBlock(&stBlock);

                        // clear the sentence
                        clearSentence(&stSentence);
                }
        }

        // if nothing else, simply add the word to the sentence
        else
        {
                addWordToSentence(&stSentence, cWordInput);
        }
    }

    apiDisconnect(fdSock);
    exit(0);
}
```

Notes

- The code has been tested with up to 3-byte encoded length. 4 and 5 byte encoded length have not been tested yet. The logic for 4-byte length should work, and 5-byte lengths are too long for standard-sized int in C.
- The code has been tested successfully with little endian (PC) and big endian (MIPSBE) processors.

API ActionScript 3 class

These are ActionScript 3 classes for working with RouterOS v3 API. You can take it, edit it and use it as you need.

NOTE - The API implementation is not yet fully tested with large requests or replies. Please update this section if you have tested this. Also, this ActionScript probably works best in Flash AIR applications (opposed to swf flash files), since they have no security limitations on socket connections.

Class

```
// ApiSocket.as
//
// RouterOS API class
// Author: Håkon Nessjøen of Avelia AS
// Date: 2. May 2009
//
package {

      import flash.errors.*;
      import flash.events.*;
      import flash.utils.ByteArray;
      import com.adobe.crypto.MD5;
      import flash.net.Socket;

      public class ApiSocket extends Socket {
            static public var RECEIVED:String = "received";
            static public var LOGIN:String = "loggedin";
            private var cmd:String;
            private var doLogin:int;
            private var user:String;
            private var password:String;
            private var returnData:Array;
            private var returnPos:int;
            private var toread:int;
            private var firstRe:int;
            private var tag:String;
            private var gotDone:Boolean;
            private var gotTrap:Boolean;
            private var gotFatal:Boolean;

            public function ApiSocket(host:String, port:uint) {
                  super(host, port);
                  toread = 0;
                  doLogin = 0;
                  addEventListener(ProgressEvent.SOCKET_DATA,
socketDataHandler);
            }
```

```
            public function login(u:String, p:String) {
                  doLogin = 1;
                  user = u;
                  password = p;
                  sendRequest("/login");
            }

            public function sendRequest(... outData):void {
                  returnData = new Array();
                  returnPos = 0;
                  firstRe = 0;
                  gotDone = false;
                  gotTrap = false;
                  gotFatal = false;
                  tag = "";
                  cmd = outData[0];
                  returnData[returnPos] = new Object();

                  for (var i:int = 0; i < outData.length; ++i) {
                        var data:ByteArray = new ByteArray();

                        var len:uint = outData[i].length;

                        if (len < 0x80)
                              data.writeByte(len);
                        else
                        if (len < 0x4000) {
                              len |= 0x8000;
                              data.writeByte((len >> 8) &
0xff);
                              data.writeByte(len & 0xff);
                        } else
                        if (len < 0x200000) {
                              len |= 0xC00000;
                              data.writeByte((len >> 16) &
0xff);
                              data.writeByte((len >> 8) &
0xff);
                              data.writeByte(len & 0xff);
                        } else
                        if (len < 0x10000000) {
                              len |= 0xE0000000;
                              data.writeByte((len >> 24) &
0xff);
                              data.writeByte((len >> 16) &
0xff);
                              data.writeByte((len >> 8) &
```

```
0xff);
                              data.writeByte(len & 0xff);
                        } else {
                              data.writeByte(0xF0);
                              data.writeByte((len >> 24) &
0xff);
                              data.writeByte((len >> 16) &
0xff);
                              data.writeByte((len >> 8) &
0xff);
                              data.writeByte(len & 0xff);
                        }

                        writeBytes(data);
                        writeUTFBytes(outData[i]);
                  }
                  writeByte(0);
                  flush();
            }

            private function readResponse():void {
                  var len:int;

                  if (toread == 0) {
                        var len1:uint = readUnsignedByte();

                        if (len1 == 0) {
                              if (gotDone || gotTrap || gotFatal) {
                                    if (doLogin == 1) {
                                          if (returnData[0].ret) {
                                                var chal:ByteArray =
new ByteArray();
                                                var md5:ByteArray = new
 ByteArray();

                                                for (var i:int = 0; i
< returnData[0].ret.length; i += 2) {

chal.writeByte(int("0x" + returnData[0].ret.substr(i,2)));
                                                }

                                                md5.writeByte(0);

md5.writeUTFBytes(password);
                                                md5.writeBytes(chal);

                                                doLogin++;
```

```
                                                // Send challenge
response
                                                sendRequest("/login",
"=name=" + user, "=response=00" + MD5.hashBytes(md5));
                                          }
                                    } else if (doLogin == 2) {
                                          doLogin = 0;
                                           dispatchEvent(new
ApiEvent(ApiSocket.LOGIN, "", returnData, gotDone ? 'done' : (gotFatal
? 'fatal' : 'trap')));
                                    } else {
                                          dispatchEvent(new
ApiEvent(ApiSocket.RECEIVED, tag, returnData, gotDone ? 'done' :
(gotFatal ? 'fatal' : 'trap')));
                                    }
                              }

                              if (bytesAvailable)
                                    readResponse();
                              else
                                    return;
                        }

                        if (len1 >= 0xF0) {
                              len = readUnsignedByte();
                              len = (len << 8) +
readUnsignedByte();
                              len = (len << 8) +
readUnsignedByte();
                              len = (len << 8) +
readUnsignedByte();
                        } else
                        if (len1 >= 0xE0) {
                              len = ((len1 & 15) << 8) +
readUnsignedByte();
                              len = (len << 8) +
readUnsignedByte();
                              len = (len << 8) +
readUnsignedByte();
                        } else
                        if (len1 >= 0xC0) {
                              len = ((len1 & 31) << 8) +
readUnsignedByte();
                              len = (len << 8) +
readUnsignedByte();
                        } else
                        if (len1 >= 0x80) {
```

```
                              len = ((len1 & 63) << 8) +
readUnsignedByte();
                        } else
                              len = len1;

                        toread = len;
                  }

                  // Calculate how much data of the full length that is
 available right now
                  var slen:int = bytesAvailable > toread ? toread :
bytesAvailable;
                  // Calculate how much data that has to be read later
                  toread = toread > bytesAvailable ? toread -
bytesAvailable : 0;

                  // Read relevant data
                  var str:String = readUTFBytes(slen);

                  if (toread == 0) {
                        if (str == '!re') {
                              firstRe++;
                              if (firstRe > 1) {
                                    returnPos++
                                    returnData[returnPos] = new
Object();
                              }
                        }
                        if (str == '!trap')
                              gotTrap = true;

                        if (str == '!fatal')
                              gotFatal = true;

                        if (str == '!done')
                              gotDone = true;

                        // Parse key-value pair
                        if (str.substr(0,1) == '=') {
                              var tmpPos:int = str.indexOf('=',1);
                              var tmpKey:String =
str.substr(1,tmpPos-1);
                              var tmpVal:String = str.substr(tmpPos+1);
                              returnData[returnPos][tmpKey] = tmpVal;
                        }

                        // Reset tag
```

```
                        if (str.substr(0,1) == '!')
                              tag = "";

                        // Set tag
                        if (str.substr(0,5) == '.tag=')
                              tag = str.substr(5);

                        // Are there more packets available
                        if (bytesAvailable)
                              readResponse();
                  }

            }

            private function
socketDataHandler(event:ProgressEvent):void {
                  readResponse();
            }
      }
}


// ApiEvent.as
//
// RouterOS API Event class
// Author: Håkon Nessjøen of Avelia AS
// Date: 2. May 2009
//
package {

      import flash.events.Event;

      public class ApiEvent extends Event {
            static public var RECEIVED:String = "received";
            static public var LOGIN:String = "loggedin";

            public var data:Array;
            public var result:String;
            public var tag:String;

            public function ApiEvent(type:String, tg:String, dta:Array,
 res:String){
                  super(type);
                  data = dta;
                  result = res;
                  tag = tg;
            }
```

```
       }

}
```

Example 1

```
// Short example that outputs wireless registration-table every second
import ApiSocket;
import ApiEvent;

var sock:ApiSocket;
var myTimer:Timer = new Timer(1000);

myButton.addEventListener(MouseEvent.CLICK, testAPI);
myTimer.addEventListener(TimerEvent.TIMER, timedFunction);

function testAPI(evt:MouseEvent):void {
      sock = new ApiSocket("172.17.1.1", 8728);
      sock.addEventListener(ApiEvent.RECEIVED, receive);
      sock.addEventListener(Event.CONNECT, connected);
      sock.addEventListener(ApiEvent.LOGIN, loggedin);
}



function connected(evt:Event) {
      trace("Connected. Logging in.");
      sock.login("admin","password");
}



function loggedin(evt:ApiEvent) {
      // result can be done, trap or fatal
      if (evt.result == 'done') {
            myTimer.start();
      }
}

function timedFunction(e:TimerEvent) {
      sock.sendRequest("/interface/wireless/registration-table/print", ".tag=mytag");
}

function receive(evt:ApiEvent) {
      trace("Got Event with tag " + evt.tag + " result: " + evt.result);

      if (evt.tag == 'mytag' && evt.result == 'done') {
            trace("Got rows: " + evt.data.length);

            for (var i:int = 0; i < evt.data.length; ++i) {
```

```
                    trace(evt.data[i]['mac-address'])
                    trace("  " + evt.data[i]['signal-strength'])
                }
        }
}
```

API Delphi Client

This is implementation of MikroTik RouterOS API Client for Delphi. It supports execution of parallel requests to router and has database-like interface for easy use.

Classes

RouterOSAPI unit contains definition of two classes which you need to work with API protocol from your Delphi programs.

TRosApiClient

This class encapsulates properties and methods to make a connection to router via RouterOS API protocol.

- function **Connect**(*Hostname*, *Username*, *Password*: String; *Port*: String = '8728'): Boolean;

This function connects to the router and performs login procedure. It returns **True** if login was successful, **False** otherwise.

- function **Query**(*Request*: array of String; *GetAllAfterQuery*: Boolean): TROSAPIResult;

Makes a query to the router. **Request** is array of string, first one being the command and others are parameters. If **GetAllAfterQuery** is **True**, then **TROSAPIResult.GetAll** is executed after sending a query.

- function **Execute**(*Request*: array of String): Boolean;

If you do not need to receive any output from your query, use this method. It simply calls **Query** function and frees returned object.

- property **Timeout**: Integer;

With this property you can set timeout value for network operations (in milliseconds).

- property **LastError**: String;

This read-only property contains textual description of last error occured.

- procedure **Disconnect**;

Disconnects from the router.

TRosApiResult

This class gives you an ability to work with data returned from queries. Each command execution is "isolated" in it's **TRosApiResult** object, so you can do parallel requests by calling **TRosApiClient.Query** and receiving several **TRosApiResult** objects.

- property **ValueByName**[*Name*: String]: String; default;

Returns the value of **Name** parameter (*word* in terms of API) in current sentence. The preferred way of getting the result is the following: **ApiResult['ParmName']** instead of **ApiResult.ValueByName('ParmName')**. You can use param name both with and without leading '=' character (*ApiResult['address']* and *ApiResult['=address']* will return the same result).

- property **Values**: TRosApiSentence;

Returns current sentence of query result (type is TRosApiSentence).

- function **GetOne**(*Wait*: Boolean): Boolean;

Receives one sentence from the router. If **Wait** parameter is **True**, function will wait until sentence is received. If **Wait** is **False** and no sentences were received for now, function returns **False**. This is helpful when executing infinite commands (like 'listen') in GUI, when you need to process other user's actions: you should periodically call **GetOne** with **Wait = False**, and in case of negative result just do something else for a time.

- function **GetAll**: Boolean;

Receives all sentences upto '!done', then returns **True** (or **False** in case of a timeout).

- property **RowsCount**: Integer;

Returns number of received sentences after calling **GetAll**.

- property **Eof**: Boolean;

Returns **True** if there's a more sentence(s) in query result.

- property **Trap**: Boolean;

Returns **True** if there were trap(s) during **GetAll**

- property **Done**: Boolean;

Returns **True** if '!done' sentence was received in **GetOne**

- procedure **Next**;

Shifts to the next sentence, received in **GetAll**

- procedure **Cancel**;

Cancels current command execution.

Examples

Sample application **APITest** you can download at Downloads and suggestions section

Creating connection to router

At first, we should declare a variable and create an instance of TRosApiClient:

```
var
  RouterOS: TRosApiClient;

RouterOS := TRosApiClient.Create;
```

Now we connect to router and perform login procedure:

```
if RouterOS.Connect('192.168.0.1', 'admin', 'password') then
begin
  //we are connected successfully
end
else
begin
  //an error occured; text error message is in LastError property
end;
```

Executing queries

All queries are done by calling **Query** function of **TRosApiClient**. It returns an instance of **TRosApiResult**, from which all data are fetched.

```
var
  Res: TRosApiResult;

Res := RouterOS.Query(['/system/resource/print'], True);
```

Obtaining the result with GetAll

```
Res := ROS.Query(['/ip/arp/print', '?interface=ether2'], True);

while not Res.Eof do
begin
  SomeProcessingFunction(Res['.id'], Res['address']);
  Res.Next;
end;

Res.Free;
```

Obtaining the result with GetOne

First, place a Timer on form and name it **tmrListen**, set **Enabled** to *False*. Then we make a query and enable timer:

```
ResListen := ROS.Query(['/log/listen'], False);
tmrListen.Enabled := True;
```

Then we check for new data on timer event:

```
procedure TForm1.tmrListenTimer(Sender: TObject);
begin
  repeat
    if not ResListen.GetOne(False) then Break;

    if ResListen.Trap then
    begin
      ShowMessage('Trap: ' + ROS.LastError);
      Break;
    end;

    if ResListen.Done then
```

```
    begin
      ShowMessage('Done');
      ResListen.Free;
      tmrListen.Enabled := False;
      Break;
    end;

    Memo1.Lines.Add(ResListen['time'] + ': ' + ResListen['message']);
  until False;
end;
```

Downloads and suggestions

For downloads and suggestions see forum thread RouterOS API Delphi Client [1]

References

[1] http://forum.mikrotik.com/viewtopic.php?f=9&t=31555&start=0

API Delphi

This document describes a Delphi class to access RouterOs using API interface.

Enable API's in RouterOs devices

By default API interface is disabled in the device, then enable it using the simple command in a terminal connection:

```
/ip service enable api
```

Using API: send commands and receive output through the socket

The mikrotik protocol to talk with api interface is well documented in the main page API. You can connect to the API interface of an RouterOs device using a TCP socket connected to 8728 port.

In this page you can found a Delphi class which encapsulate the details of the connection and give you the ability to use some simple methods to create applications usin API.

All methods return 0 if the execution is correct (<0 otherwise)

The Delphi class tr_mkrouter

This class is defined in a Delphi unit. I used Delphi 7 but is simple to adapt to other versions. This class publish this methods:

constructor create(i_logger: TLogger);

Create the object. Passing the object TLogger, enable you to log the activity to a memo control and/or to a file.

function open(ip_router, user, password: string): integer;

Open the socket with the router and executes the login handshake.

function msend(vs: string; fl_execute: boolean): integer;

Send the string *vs* tho the router (send an api word). If fl_execute=true it concatenate a #0 (send an api sentence) wich cause the command to be executed.

function send_command(cmd_arr: array of string): integer;

Send the strings contained in the array, then send a #0 (example: *res:=send_command(['/ppp/active/print','=stats=','=without-paging=']);*). Each send_command increase the command counter of the object. Each !done decrease this counter. You can test if there are pending commands calling the following method: **function tr_mkrouter.command_pending: boolean;**

function mrecv_sentence(var vs: string): integer;

Wait for output from router, in the vs string return the text until #0 is received. It receive an api sentence.

function mrecv_done(var vs: string): integer;

Wait for output from router, in the *vs* string return the text until *!done* is received. It receive the full response of a command.

function query_router(cmd_arr: array of string; var res: string): integer;

Is the union of a send_command and a mrecv_done.

Build the project: the libraries used

The download, contains this libraries:

- MD5.pas class (form Francois Piette), used in login handshake, at MD5.PAS [1]

- Synapse library to work with sockets at Synapse [2]

- Logger library: class used to log in a memo and/or in a file the application activity

- Utils library: contain some useful procedures to work with IP and strings

- Synapse library to work with sockets at Synapse [2]

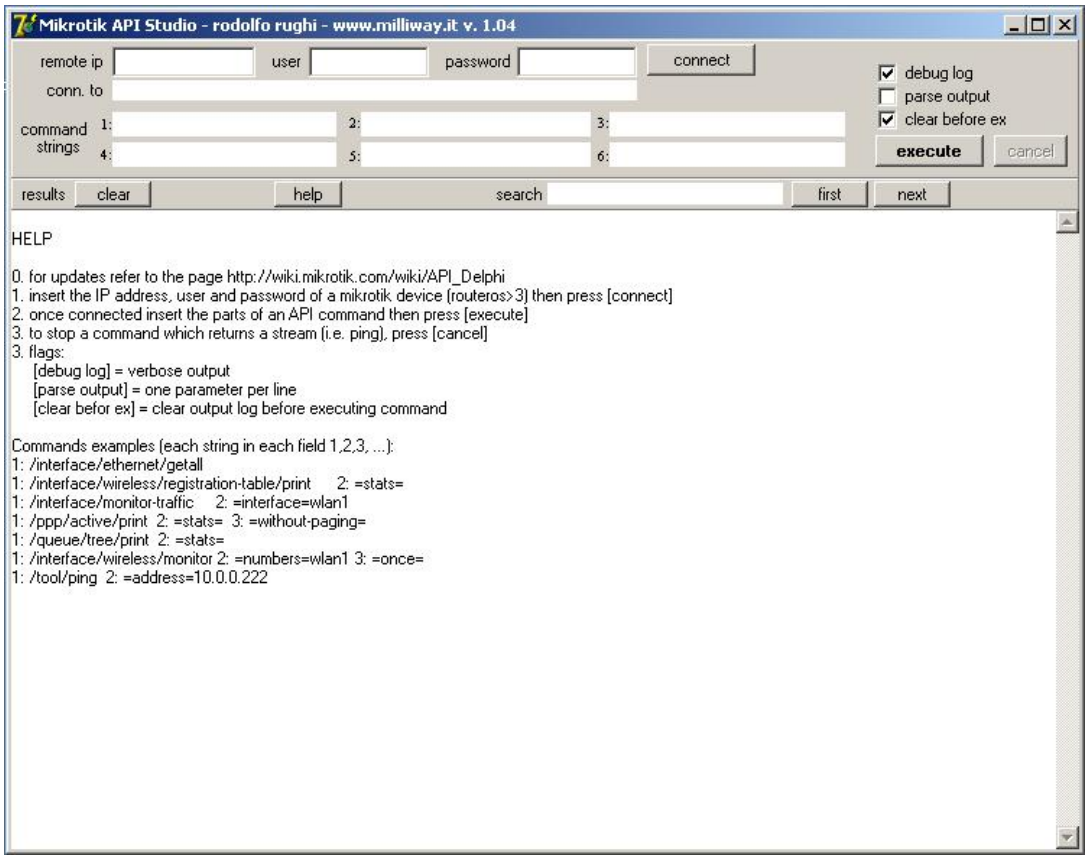
To build the sample application, add the word "synapse" to the Project>Options>Directory Conditionals>Search path.

In different versions of delphi, could be necessary to modify some links to used units (ex. is you use Delphi 5, you must provide StrUtil.pas (thanks to pedja) at strutil.pas [3])

The API_STUDIO application

I usually develop using mikrotik API and the *API STUDIO* allow me to test API commands before implementing it in other software.

This application is very simple but useful and is a demo for this delphi class (you can also read the simple help in the main form).



LOGIN

- Insert login informations of router you want to test then print **connect**.

- The objet of class tr_mkrouter is created and **open** method is called.

- If the router is connected, api studio perform a /system/identity/getall to retrieve the identity of the router.

EXECUTE COMMANDS

- Insert the command you want to send to the router, each word in a field.

- By pressing **execute** button the **send_command** method is called and a timer is activated.

- The timer call the method **mrecv_sentence** and show results until **command_pending** returns false.

- You can break the result stream (example, in case you have asked for a ping) by pressing the **cancel** button. This button call a send_command(['/cancel']) (this increases the number of pending commands); you could wait for a while before the stream stops because the timer must receive the !done for the previous command and the !done for the /cancel.

DOWNLOADS

Please refer to this forum thread for download: API_STUDIO [4]. The download contain all source code and the executable (compiled for i386) of the client to test api commands.

References

[1] http://www.koders.com/delphi/fid5A4F925F646C191A79107D11EDD80DDDF205615E.aspx?s=md5
[2] http://www.ararat.cz/synapse/doku.php/download
[3] http://www.koders.com/delphi/fidDF48A5F25F06E3C6B1419E0691B806FF60260646.aspx?s=delphi
[4] http://forum.mikrotik.com/viewtopic.php?f=9&t=28821

API in C Sharp

This is C# class for connecting and working with v3.x API

Class

```
class MK

    {

        Stream connection;

        TcpClient con;


        public MK(string ip)

        {

            con = new TcpClient();

            con.Connect(ip, 8728);

            connection = (Stream)con.GetStream();

        }

        public void Close()

        {

            connection.Close();

            con.Close();

        }

        public bool Login(string username, string password)

        {

            Send("/login", true);

            string hash = Read()[0].Split(new string[] { "ret=" }, StringSplitOptions.None)[1];

            Send("/login");

            Send("=name=" + username);

            Send("=response=00" + EncodePassword(password, hash), true);

            if (Read()[0] == "!done")

            {

                return true;

            }

            else

            {

                return false;

            }
```

```
        }
        public void Send(string co)
        {
            byte[] bajty = Encoding.ASCII.GetBytes(co.ToCharArray());
            byte[] velikost = EncodeLength(bajty.Length);

            connection.Write(velikost, 0, velikost.Length);
            connection.Write(bajty, 0, bajty.Length);
        }
        public void Send(string co, bool endsentence)
        {
            byte[] bajty = Encoding.ASCII.GetBytes(co.ToCharArray());
            byte[] velikost = EncodeLength(bajty.Length);
            connection.Write(velikost, 0, velikost.Length);
            connection.Write(bajty, 0, bajty.Length);
            connection.WriteByte(0);
        }
        public List<string> Read()
        {
            List<string> output = new List<string>();
            string o = "";
            byte[] tmp = new byte[4];
            long count;
            while (true)
            {
                tmp[3] = (byte)connection.ReadByte();
                //if(tmp[3] == 220) tmp[3] = (byte)connection.ReadByte(); it sometimes happend to me that
                //mikrotik send 220 as some kind of "bonus" between words, this fixed things, not sure about it though
                if (tmp[3] == 0)
                {
                    output.Add(o);
                    if (o.Substring(0, 5) == "!done")
                    {
                        break;
                    }
                    else
                    {
                        o = "";
                        continue;
                    }
                }
                else
                {
                    if (tmp[3] < 0x80)
                    {
                        count = tmp[3];
                    }
```

```
                else
                {
                    if (tmp[3] < 0xC0)
                    {
                        int tmpi = BitConverter.ToInt32(new byte[] { (byte)connection.ReadByte(), tmp[3],0,0 }, 0);
                        count = tmpi ^ 0x8000;
                    }
                    else
                    {
                        if (tmp[3] < 0xE0)
                        {
                            tmp[2] = (byte)connection.ReadByte();
                            int tmpi = BitConverter.ToInt32(new byte[] { (byte)connection.ReadByte(), tmp[2], tmp[3],0 }, 0);
                            count = tmpi ^ 0xC00000;
                        }
                        else
                        {
                            if (tmp[3] < 0xF0)
                            {
                                tmp[2] = (byte)connection.ReadByte();
                                tmp[1] = (byte)connection.ReadByte();
                                int tmpi = BitConverter.ToInt32(new byte[] { (byte)connection.ReadByte(), tmp[1], tmp[2], tmp[3] }, 0);
                                count = tmpi ^ 0xE0000000;
                            }
                            else
                            {
                                if (tmp[3] == 0xF0)
                                {
                                    tmp[3] = (byte)connection.ReadByte();
                                    tmp[2] = (byte)connection.ReadByte();
                                    tmp[1] = (byte)connection.ReadByte();
                                    tmp[0] = (byte)connection.ReadByte();
                                    count = BitConverter.ToInt32(tmp, 0);
                                }
                                else
                                {
                                    //Error in packet reception, unknown length
                                    break;
                                }
                            }
                        }
                    }
                }
            }


            for (int i = 0; i < count; i++)
            {
```

```
                o += (Char)connection.ReadByte();
            }
        }
        return output;
    }
    byte[] EncodeLength(int delka)
    {
        if (delka < 0x80)
        {
            byte[] tmp = BitConverter.GetBytes(delka);
            return new byte[1] { tmp[0] };
        }
        if (delka < 0x4000)
        {
            byte[] tmp = BitConverter.GetBytes(delka | 0x8000);
            return new byte[2] { tmp[1], tmp[0] };
        }
        if (delka < 0x200000)
        {
            byte[] tmp = BitConverter.GetBytes(delka | 0xC00000);
            return new byte[3] { tmp[2], tmp[1], tmp[0] };
        }
        if (delka < 0x10000000)
        {
            byte[] tmp = BitConverter.GetBytes(delka | 0xE0000000);
            return new byte[4] { tmp[3], tmp[2], tmp[1], tmp[0] };
        }
        else
        {
            byte[] tmp = BitConverter.GetBytes(delka);
            return new byte[5] { 0xF0, tmp[3], tmp[2], tmp[1], tmp[0] };
        }
    }

    public string EncodePassword(string Password, string hash)
    {
        byte[] hash_byte = new byte[hash.Length / 2];
        for (int i = 0; i <= hash.Length - 2; i += 2)
        {
            hash_byte[i / 2] = Byte.Parse(hash.Substring(i, 2), System.Globalization.NumberStyles.HexNumber);
        }
        byte[] heslo = new byte[1 + Password.Length + hash_byte.Length];
        heslo[0] = 0;
        Encoding.ASCII.GetBytes(Password.ToCharArray()).CopyTo(heslo, 1);
        hash_byte.CopyTo(heslo, 1 + Password.Length);

        Byte[] hotovo;
```

```
            System.Security.Cryptography.MD5 md5;


            md5 = new System.Security.Cryptography.MD5CryptoServiceProvider();


            hotovo = md5.ComputeHash(heslo);


            //Convert encoded bytes back to a 'readable' string
            string navrat = "";
            foreach (byte h in hotovo)
            {
                navrat += h.ToString("x2");
            }
            return navrat;
        }
    }
```

Example

```
using System.IO;
using System.Net.Sockets;


class Program
    {
        static void Main(string[] args)
        {

            MK mikrotik = new MK("your ip here");
            if (!mikrotik.Login("username", "password"))
            {
                Console.WriteLine("Could not log in");
                mikrotik.Close();
                return;
            }
            mikrotik.Send("/system/identity/getall");
            mikrotik.Send(".tag=sss", true);
            foreach (string h in mikrotik.Read())
            {
                Console.WriteLine(h);
            }
            Console.ReadKey();
        }
    }
```

Example 2

Block SMTP port and specific Ip rule by Oguzhan

```
using System.IO;
using System.Net.Sockets;

    class Program
    {
        static void Main(string[] args)
        {
            string ip = args[0];
            MK mikrotik = new MK("your ip here");
            if (mikrotik.Login("admin", "P@ssW0rd"))
            {
                mikrotik.Send("/ip/firewall/filter/add");
                mikrotik.Send("=action=drop");
                mikrotik.Send("=chain=forward");
                mikrotik.Send("=dst-port=25");
                mikrotik.Send("=protocol=tcp");
                mikrotik.Send("=protocol=tcp");
                mikrotik.Send(String.Format("=src-address={0}",ip));

                mikrotik.Send(".tag=firewall", true);

                foreach (string h in mikrotik.Read())
                {
                    Console.WriteLine(h);
                }
            }
        }
    }
```

Notes

- I have not tested it thorougly (especialy length encoding with longer words)
- You have to have using System.IO; and using System.Net.Sockets;
- Exceptions are not handled

API PHP class

This is PHP class for working with RouterOS API. You can take it, edit it and use it as you need.

NOTE - The class as shown does not work for large replies

Author

Denis Basta (Denis [dot] Basta [at] gmail [dot] com)

Contributors

Nick Barnes
Ben Menking (ben [at] infotechsc [dot] com)
Jeremy Jefferson (http://jeremyj.com)
Cristian Deluxe (djcristiandeluxe [at] gmail [dot] com)

Changelog

1.0 **Denis Basta** (Denis [dot] Basta [at] gmail [dot] com) First PHP Class released from author

1.1 **Nick Barnes**
read() function altered to take into account the placing of the "!done" reply and also correct calculation of the reply length.

1.2 **Ben Menking** (ben [at] infotechsc [dot] com)
read() function altered removed echo statement that dumped byte data to screen

1.3 **Jeremy Jefferson** (http://jeremyj.com)
January 8, 2010
Fixed write function in order to allow for queries to be executed

1.4 **Cristian Deluxe** (djcristiandeluxe [at] gmail [dot] com)
November 17, 2011
comm() function altered, added the possibility of make regular exp queries.
parse_response() and parse_response4smarty() functions altered to support a "single data" responses from server
Added documentation to functions following PHPDoc guidelines
Added version number (1.4) for follow the changes more easy

1.5 **Cristian Deluxe** (djcristiandeluxe [at] gmail [dot] com)
October 6, 2013
[Fix] Uninitialized variable error in function "read"
The last edit by user "Eugenevdm" causes a syntax error (missing $ before variable name)
Thanks also to Daniel Machado for report this bug.
[Fix] Accidental assignment in a condition in function "connect"
[Fix] Uninitialized variable in function "connect"
[Fix] Remove unused variable from "parse_response" function
[Fix] Uninitialized variable in function "parse_response"
Updated version number to 1.5

Class

```
<?php

/******************************

 *

 * RouterOS PHP API class v1.5

 * Author: Denis Basta

 * Contributors:

 *    Nick Barnes

 *    Ben Menking (ben [at] infotechsc [dot] com)

 *    Jeremy Jefferson (http://jeremyj.com)

 *    Cristian Deluxe (djcristiandeluxe [at] gmail [dot] com)

 *

 * http://www.mikrotik.com

 * http://wiki.mikrotik.com/wiki/API_PHP_class

 *

 ******************************/


class routeros_api

{

    var $debug = false;      // Show debug information

    var $error_no;           // Variable for storing connection error number, if any

    var $error_str;          // Variable for storing connection error text, if any

    var $attempts = 5;       // Connection attempt count

    var $connected = false;  // Connection state

    var $delay = 3;          // Delay between connection attempts in seconds

    var $port = 8728;        // Port to connect to

    var $timeout = 3;        // Connection attempt timeout and data read timeout

    var $socket;             // Variable for storing socket resource


    /**

     * Print text for debug purposes

     *

     * @param string      $text       Text to print

     *

     * @return void

     */

    function debug($text)

    {

        if ($this->debug)

            echo $text . "\n";

    }




    /**

     *

     *

     * @param string        $length
```

```
     *
     * @return void
     */
    function encode_length($length)
    {
        if ($length < 0x80) {
            $length = chr($length);
        } else if ($length < 0x4000) {
            $length |= 0x8000;
            $length = chr(($length >> 8) & 0xFF) . chr($length & 0xFF);
        } else if ($length < 0x200000) {
            $length |= 0xC00000;
            $length = chr(($length >> 16) & 0xFF) . chr(($length >> 8) & 0xFF) . chr($length & 0xFF);
        } else if ($length < 0x10000000) {
            $length |= 0xE0000000;
            $length = chr(($length >> 24) & 0xFF) . chr(($length >> 16) & 0xFF) . chr(($length >> 8) & 0xFF) . chr($length & 0xFF);
        } else if ($length >= 0x10000000)
            $length = chr(0xF0) . chr(($length >> 24) & 0xFF) . chr(($length >> 16) & 0xFF) . chr(($length >> 8) & 0xFF) . chr($length & 0xFF);
        return $length;
    }



    /**
     * Login to RouterOS
     *
     * @param string      $ip         Hostname (IP or domain) of the RouterOS server
     * @param string      $login      The RouterOS username
     * @param string      $password   The RouterOS password
     *
     * @return boolean                If we are connected or not
     */
    function connect($ip, $login, $password)
    {
        for ($ATTEMPT = 1; $ATTEMPT <= $this->attempts; $ATTEMPT++) {
            $this->connected = false;
            $this->debug('Connection attempt #' . $ATTEMPT . ' to ' . $ip . ':' . $this->port . '...');
            $this->socket = @fsockopen($ip, $this->port, $this->error_no, $this->error_str, $this->timeout);
            if ($this->socket) {
                socket_set_timeout($this->socket, $this->timeout);
                $this->write('/login');
                $RESPONSE = $this->read(false);
                if ($RESPONSE[0] == '!done') {
                    $MATCHES = array();
                    if (preg_match_all('/[^=]+/i', $RESPONSE[1], $MATCHES)) {
                        if ($MATCHES[0][0] == 'ret' && strlen($MATCHES[0][1]) == 32) {
                            $this->write('/login', false);
                            $this->write('=name=' . $login, false);
```

```
                        $this->write('=response=00' . md5(chr(0) . $password . pack('H*', $MATCHES[0][1])));
                        $RESPONSE = $this->read(false);
                        if ($RESPONSE[0] == '!done') {
                            $this->connected = true;
                            break;
                        }
                    }
                }
            }
            fclose($this->socket);
        }
        sleep($this->delay);
    }
    if ($this->connected)
        $this->debug('Connected...');
    else
        $this->debug('Error...');
    return $this->connected;
}



/**
 * Disconnect from RouterOS
 *
 * @return void
 */
function disconnect()
{
    fclose($this->socket);
    $this->connected = false;
    $this->debug('Disconnected...');
}



/**
 * Parse response from Router OS
 *
 * @param array       $response   Response data
 *
 * @return array                  Array with parsed data
 */
function parse_response($response)
{
    if (is_array($response)) {
        $PARSED      = array();
        $CURRENT     = null;
        $singlevalue = null;
```

```
        foreach ($response as $x) {
            if (in_array($x, array(
                '!fatal',
                '!re',
                '!trap'
            ))) {
                if ($x == '!re') {
                    $CURRENT =& $PARSED[];
                } else
                    $CURRENT =& $PARSED[$x][];
            } else if ($x != '!done') {
                $MATCHES = array();
                if (preg_match_all('/[^=]+/i', $x, $MATCHES)) {
                    if ($MATCHES[0][0] == 'ret') {
                        $singlevalue = $MATCHES[0][1];
                    }
                                $CURRENT[$MATCHES[0][0]] = (isset($MATCHES[0][1]) ? $MATCHES[0][1] : '');
                            }
            }
        }
        if (empty($PARSED) && !is_null($singlevalue)) {
            $PARSED = $singlevalue;
        }
        return $PARSED;
    } else
        return array();
}



/**
 * Parse response from Router OS
 *
 * @param array       $response   Response data
 *
 * @return array                  Array with parsed data
 */
function parse_response4smarty($response)
{
    if (is_array($response)) {
        $PARSED  = array();
        $CURRENT = null;
        $singlevalue = null;
        foreach ($response as $x) {
            if (in_array($x, array(
                '!fatal',
                '!re',
                '!trap'
```

```
            ))) {
                if ($x == '!re')
                    $CURRENT =& $PARSED[];
                else
                    $CURRENT =& $PARSED[$x][];
            } else if ($x != '!done') {
                $MATCHES = array();
                if (preg_match_all('/[^=]+/i', $x, $MATCHES)) {
                    if ($MATCHES[0][0] == 'ret') {
                        $singlevalue = $MATCHES[0][1];
                    }
                    $CURRENT[$MATCHES[0][0]] = (isset($MATCHES[0][1]) ? $MATCHES[0][1] : '');
                          }
            }
        }
        foreach ($PARSED as $key => $value) {
            $PARSED[$key] = $this->array_change_key_name($value);
        }
        return $PARSED;
        if (empty($PARSED) && !is_null($singlevalue)) {
            $PARSED = $singlevalue;
        }
    } else {
        return array();
    }
}



/**
 * Change "-" and "/" from array key to "_"
 *
 * @param array       $array      Input array
 *
 * @return array                  Array with changed key names
 */
function array_change_key_name(&$array)
{
    if (is_array($array)) {
        foreach ($array as $k => $v) {
            $tmp = str_replace("-", "_", $k);
            $tmp = str_replace("/", "_", $tmp);
            if ($tmp) {
                $array_new[$tmp] = $v;
            } else {
                $array_new[$k] = $v;
            }
        }
```

```
            return $array_new;
        } else {
            return $array;
        }
    }


    /**
     * Read data from Router OS
     *
     * @param boolean     $parse      Parse the data? default: true
     *
     * @return array                  Array with parsed or unparsed data
     */
    function read($parse = true)
    {
        $RESPONSE = array();
        $receiveddone = false;
        while (true) {
            // Read the first byte of input which gives us some or all of the length
            // of the remaining reply.
            $BYTE   = ord(fread($this->socket, 1));
            $LENGTH = 0;
            // If the first bit is set then we need to remove the first four bits, shift left 8
            // and then read another byte in.
            // We repeat this for the second and third bits.
            // If the fourth bit is set, we need to remove anything left in the first byte
            // and then read in yet another byte.
            if ($BYTE & 128) {
                if (($BYTE & 192) == 128) {
                    $LENGTH = (($BYTE & 63) << 8) + ord(fread($this->socket, 1));
                } else {
                    if (($BYTE & 224) == 192) {
                        $LENGTH = (($BYTE & 31) << 8) + ord(fread($this->socket, 1));
                        $LENGTH = ($LENGTH << 8) + ord(fread($this->socket, 1));
                    } else {
                        if (($BYTE & 240) == 224) {
                            $LENGTH = (($BYTE & 15) << 8) + ord(fread($this->socket, 1));
                            $LENGTH = ($LENGTH << 8) + ord(fread($this->socket, 1));
                            $LENGTH = ($LENGTH << 8) + ord(fread($this->socket, 1));
                        } else {
                            $LENGTH = ord(fread($this->socket, 1));
                            $LENGTH = ($LENGTH << 8) + ord(fread($this->socket, 1));
                            $LENGTH = ($LENGTH << 8) + ord(fread($this->socket, 1));
                            $LENGTH = ($LENGTH << 8) + ord(fread($this->socket, 1));
                        }
                    }
```

```
            }
        } else {
            $LENGTH = $BYTE;
        }
        // If we have got more characters to read, read them in.
        if ($LENGTH > 0) {
            $_      = "";
            $retlen = 0;
            while ($retlen < $LENGTH) {
                $toread = $LENGTH - $retlen;
                $_ .= fread($this->socket, $toread);
                $retlen = strlen($_);
            }
            $RESPONSE[] = $_;
            $this->debug('>>> [' . $retlen . '/' . $LENGTH . '] bytes read.');
        }
        // If we get a !done, make a note of it.
        if ($_ == "!done")
            $receiveddone = true;
        $STATUS = socket_get_status($this->socket);
        if ($LENGTH > 0)
            $this->debug('>>> [' . $LENGTH . ', ' . $STATUS['unread_bytes'] . ']' . $_);
        if ((!$this->connected && !$STATUS['unread_bytes']) || ($this->connected && !$STATUS['unread_bytes'] && $receiveddone))
            break;
    }
    if ($parse)
        $RESPONSE = $this->parse_response($RESPONSE);
    return $RESPONSE;
}



/**
 * Write (send) data to Router OS
 *
 * @param string      $command    A string with the command to send
 * @param mixed       $param2     If we set an integer, the command will send this data as a "tag"
 *                                If we set it to boolean true, the funcion will send the comand and finish
 *                                If we set it to boolean false, the funcion will send the comand and wait for next command
 *                                Default: true
 *
 * @return boolean                Return false if no command especified
 */
function write($command, $param2 = true)
{
    if ($command) {
        $data = explode("\n", $command);
        foreach ($data as $com) {
```

```
                $com = trim($com);
                fwrite($this->socket, $this->encode_length(strlen($com)) . $com);
                $this->debug('<<< [' . strlen($com) . '] ' . $com);
            }
            if (gettype($param2) == 'integer') {
                fwrite($this->socket, $this->encode_length(strlen('.tag=' . $param2)) . '.tag=' . $param2 . chr(0));
                $this->debug('<<< [' . strlen('.tag=' . $param2) . '] .tag=' . $param2);
            } else if (gettype($param2) == 'boolean')
                fwrite($this->socket, ($param2 ? chr(0) : ''));
            return true;
        } else
            return false;
    }


    /**
     * Write (send) data to Router OS
     *
     * @param string      $com        A string with the command to send
     * @param array       $arr        An array with arguments or queries
     *
     * @return array                  Array with parsed
     */
    function comm($com, $arr = array())
    {
        $count = count($arr);
        $this->write($com, !$arr);
        $i = 0;
        foreach ($arr as $k => $v) {
            switch ($k[0]) {
                case "?":
                    $el = "$k=$v";
                    break;
                case "~":
                    $el = "$k~$v";
                    break;
                default:
                    $el = "=$k=$v";
                    break;
            }
            $last = ($i++ == $count - 1);
            $this->write($el, $last);
        }
        return $this->read();
    }
}
?>
```

Example 1

```
<?php

require('routeros_api.class.php');

$API = new routeros_api();

$API->debug = true;

if ($API->connect('111.111.111.111', 'LOGIN', 'PASSWORD')) {

   $API->write('/interface/getall');

   $READ = $API->read(false);
   $ARRAY = $API->parse_response($READ);

   print_r($ARRAY);

   $API->disconnect();

}

?>
```

OR

```
<?php

require('routeros_api.class.php');

$API = new routeros_api();

$API->debug = true;

if ($API->connect('111.111.111.111', 'LOGIN', 'PASSWORD')) {

   $API->write('/interface/getall');
   $ARRAY = $API->read();

   print_r($ARRAY);

   $API->disconnect();

}

?>
```

OR

```
<?php

require('routeros_api.class.php');

$API = new routeros_api();

$API->debug = true;

if ($API->connect('111.111.111.111', 'LOGIN', 'PASSWORD')) {

   $ARRAY = $API->comm('/interface/getall');
   print_r($ARRAY);

   $API->disconnect();

}

?>
```

Output

```
Array
(
   [0] => Array
      (
         [.id] => *1
         [name] => ether1
         [mtu] => 1500
         [type] => ether
         [running] => yes
         [dynamic] => no
         [slave] => no
         [comment] =>
         [disabled] => no
      )

   [1] => Array
      (
         [.id] => *2
         [name] => ether2
         [mtu] => 1500
         [type] => ether
         [running] => yes
         [dynamic] => no
         [slave] => no
         [comment] =>
         [disabled] => no
      )
```

```
   [2] => Array
      (
         [.id] => *3
         [name] => ether3
         [mtu] => 1500
         [type] => ether
         [running] => yes
         [dynamic] => no
         [slave] => no
         [comment] => ether3
         [disabled] => no
      )
)
```

Example 2

Thanks a lot for this API, It help me a lot to write my php page for our support team to have access to wireless registration table.

```
 $API->write('/interface/wireless/registration-table/print',false);
 $API->write('=stats=');
```

Output

```
Array
(
[0] =>
   Array
   (
   [.id] => *147
   [comment] =>
   [interface] => AP101
   [mac-address] => 00:0B:6B:37:58:33
   [ap] => false
   [wds] => false
   [rx-rate] => 11Mbps
   [tx-rate] => 11Mbps
   [packets] => 4043966,2669114
   [bytes] => 3961713942,280551024
   [frames] => 4043966,2669114
   [frame-bytes] => 3937477458,264536340
   [hw-frames] => 4500839,2669114
   [hw-frame-bytes] => 256326637,349947988
   [tx-frames-timed-out] => 0
   [uptime] => 1w13:09:12
   [last-activity] => 00:00:00.090
   [signal-strength] => -73dBm@11Mbps
   [signal-to-noise] => 30
```

```
   [strength-at-rates] => -73dBm@1Mbps 4m4s640ms,-73dBm@2Mbps 4m58s730ms,-73dBm@5.5Mbps 42s450ms,-73dBm@11Mbps 90ms
   [tx-ccq] => 91
   [p-throughput] => 5861
   [ack-timeout] => 31
   [last-ip] => 192.168.0.220
   [802.1x-port-enabled] => true
   [wmm-enabled] => false
   )
[1] => Array
   (
       ...
    )
...
)
```

Example 3

Adding vpn user

```
$API->comm("/ppp/secret/add", array(
           "name"     => "user",
           "password" => "pass",
           "remote-address" => "172.16.1.10",
           "comment"  => "{new VPN user}",
           "service"  => "pptp",
));
```

Example 4

Find registration-table id for specified MAC

```
$ARRAY = $API->comm("/interface/wireless/registration-table/print", array(
  ".proplist"=> ".id",
  "?mac-address" => "00:0E:BB:DD:FF:FF",
));
print_r($ARRAY);
```

Example 5

Count leases from specific IP Pool (using regexp count all IPs starting with 1.1.x.x)

```
$ARRAY = $API->comm("/ip/dhcp-server/lease/print", array(
    "count-only"=> "",
    "~active-address" => "1.1.",
));
print_r($ARRAY);
```

or

```
$API->write('/ip/dhcp-server/lease/print', false);
$API->write('=count-only=', false);
```

```
$API->write('~active-address~"1.1."');
$ARRAY = $API->read();
print_r($ARRAY);
```

Returns a number with leases

MikroTik for Mac

There are several possibilities to use Winbox on Apple Mac computers:

1. Winbox wrapper [1] v1.4 by Ivan Tiukov (ivan@tiukov.com)
2. Winbox in Winebottler by SomniusX (hellasproject.com)
3. Install your own Wine (see below)

Darwine

To use *WinBox* under Mac OS X (possible on Intel-based Macs *only*), you have to install *Darwine*, a port of Wine and other supporting tools that allows Darwin and Mac OS X users to run Windows applications. Wine is an Open Source implementation of the Windows API on top of X and Unix.

Installation instructions for Mac OS X Leopard 10.5

You can use a combined Darwine package that includes fonts and creates the symlinks. Read about it and download the latest version here: http://thisismyinter.net/?p=47

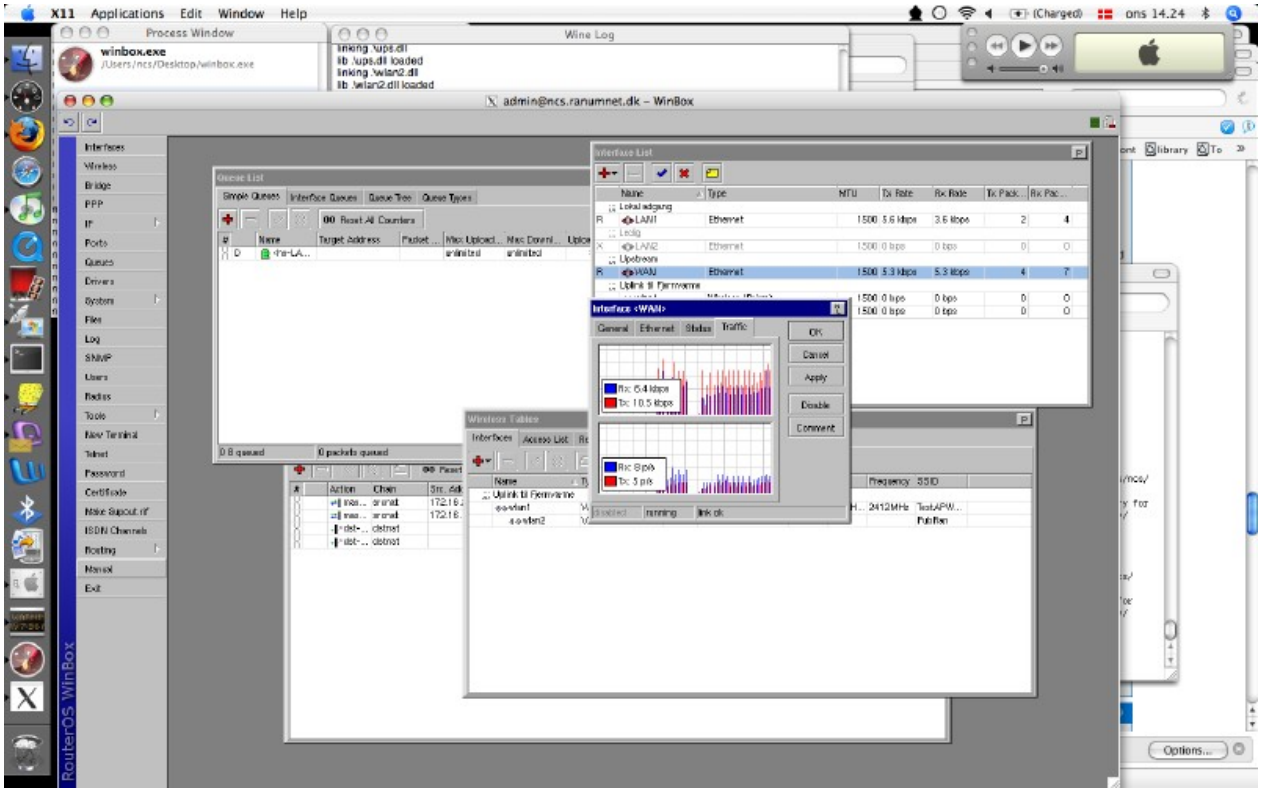
Once installed, simply run Winbox.exe from anywhere.

Darwine installation instructions on Mac OS X Tiger 10.4

It is possible to get WinBox running on Intel based Mac's running Mac OS X Tiger (10.4) however there are a few things that are important:

1. Install X11. It's on the DVD that came with your Mac.
2. Install XCode. It's on the same DVD as X11.
3. Download Darwine for Tiger from http://thisismyinter.net/?p=29 or http://www.kronenberg.org/darwine/- Both of these builds have FreeType built in, however you should still install [2] if you use the build from Kronenberg.org.

After this, you should be able to run Winbox.exe from wherever you wish to place it.



MacPorts and wine

If you cannot get Darwine to work, you can install Wine using Mac Ports at http://guide.macports.org/ Once MacPorts is installed open terminal and type the following:

```
# sudo port install wine
```

This will take quite a while to download and compile all the relevant packages needed. When finished you can type the following into a terminal window to start Winbox. I'm assuming here that you copied a version of winbox.exe to the /Applications folder

Note For Snow Leopard Users in 64 bit mode:

- install wine-devel instead of wine
- as for Nov 2009, there is a bug in wine-devel [3], so don't forget to add +universal when installing wine.
- the correct command will be: sudo port install wine-devel +universal

```
# /opt/local/bin/wine /Applications/winbox.exe
```

If that works then you are good to go. As a little plus you can download the following automator application that will run the above piece of script but with the benefit of being an application. You can just double click on it to start winbox. The file is at http://www.mediafire.com/file/3wdywhmmnhy/Winbox.zip.Unzip and view in automator if you want to check the contents before running.

Making Home/End/PgUp/PgDown work in Terminal.app

It really annoys me that by default you have no Home/End functionality in terminal.app. Editing large RouterOS scripts or just trying to modify already typed command can become a tedious task without these. So here is a quick guide how to fix this.

1. Open Terminal.app
2. Go to Terminal->Window settings
3. Choose "Keyboard" from the drop down menu
4. Set action of the "home", "end", "page down" and "page up" keys to "send string to shell" and set the string the following values:

```
:: end -- "<esc>[4~"
:: home -- "<esc>[1~"
:: pg up -- "<esc>[5~"
:: pg down -- "<esc>[6~"
```

<esc> -- means you have to actually press the [esc] key

How to capture RouterOS sniffer stream on Mac

To capture MT sniffer stream on Mac you can use Wireshark [4]. For instructions on how to set up sniffing with Wireshark, see Ethereal/Wireshark

Mac-How [5] /Eugene

References

[1] http://dl.dropbox.com/u/3669437/Winbox_1.4.dmg
[2] http://fontforge.sourceforge.net/lFontForge
[3] http://trac.macports.org/ticket/21865
[4] http://www.wireshark.org/
[5] http://www.mac-how.net

Assorted examples

- Automated Backups
- Automatic Backup with Centralized Storage - By Ashish Patel
- Bash scripts for Linux/Mysql/Freeradius/PPPoE
- Booting RouterOS from USB drives
- Centralized Authentication for Hotspot user
- Change MAC address of VLAN interface
- Configuring RouterOS to work with EVDO/3G PCMCIA card
- Configuring RouterOS to work with EVDO/3G USB Modem
- Event WiFi By EITL-India
- External Squid Box with No Limit Cache HIT Object ROS 2.9
- Hotspots all over town with one User Manger
- How to block MSN Messenger
- How to connect a 2Wire BT ADSL Router to RouterOS
- How to configure a home router
- How to Connect your Home Network to xDSL Line
- How to connect your office network with Dial-up Modem/Connectivity
- How to restrict Lan-Wan but keep Lan-Lan unrestricted using only simple queues.
- How to link Public addresses to Local ones
- How to make a HotSpot gateway
- How to make transparent web proxy
- How To Manage RouterOS Using Webmin
- How to setup up RADIUS for use with MikroTik - By Ramona
- How to translate hotspot interface
- Internet in Campus/Township using EPABX ,Mikrotik with MoxaC168H & Dialup-Modems -By Prashant Limbachia
- Monitoring Network thru SMS Alerts
- Several Web Servers using one public IP address
- Reverting from Test packages to normal (x86)
- Reverting from Test packages to normal (rb500)
- RouterOS & Radius For PPPoE - Full Guide
- Using RouterOS to send SMS messages
- TeamSpeak spam protection
- Using SSH for system backup
- Use HOTSPOT for advertisement purpose
- Virtualization
- Setup local NTP servers
- Use Cisco ACS to manage Logins and Permissions by Group
- Private Management Network with Vlans Using minimial Configuration
- Tutorials in serbian language
- Tutorials in spanish language

Article Sources and Contributors

MikroTik RouterOS *Source*: http://wiki.mikrotik.com/index.php?oldid=22355 *Contributors*: Dzintars, Eugene, HarvSki, Lastguru, Marisb, MediaWiki default, Normis, Paulhoff, Rainer Wieland, Rieks, Route, WikiSysop

Hardware *Source*: http://wiki.mikrotik.com/index.php?oldid=20098 *Contributors*: Brianlewis, Cmit, DonGould, Marisb, Normis, Rajeshrouthu, Sdischer

Supported Hardware *Source*: http://wiki.mikrotik.com/index.php?oldid=25936 *Contributors*: Adjoneidi, Airnet, Airstream, Aivas, Ajar, Albarnaz, Alex.alouit, Alexspils, Andreacoppini, Anson99, Becs, BluThunder, Brianlewis, Bs85, Butche, Cdiggity, Changeip, Chrone, Chronos, Clear cmos, Clmanning, Compex, Daniel.szilagyi, Dankerr, Dcsindo, Dental material, Diginet, Digitalnet, Dingram, Dxinfo, Eben, Eigza, ElBerto, Enk, Fbaffoni, Felipescu, Femur, Funky, GRiffiN, Galaxy, Gcakici, Gerard, Gmeden, Gmsmstr, Grizz, Gustkiller, Hegars, Hilton, Hocimin, Iam8up, Ibersystems, Insidertech, Interlink, Itsh.net, JJOliver998, Janisk, Jens18, Jfraga, Jjcinaz, Jjnz1, Jorgeamaral, Jorj, Jp, Jreames, Jtmr, Kappi, Kostil, Krisjanis, Krogalin, Kryptyk, Laurinkus, Leoktv, Magnus, Marisb, Mariusbm, Marlow, Maximan, Mdown, Megis, Meverest, Micah, Mlundahl, Mr.BS, Nborson, Nbright, Nest, Netrat, Nettraptor, NetworkPro, Ngwill, Nhickman, Normis, Npero, Nzl, Pelish, Priidik, QpoX, Raf, Rafaelenrike, Ratbaggy, Remyliaw, Rieks, Rjickity, Rjscomms, Robbar, Roc-noc.com, RogerWilco, RubahFox, Scampbell, Scream, Seacol, Sergejs, SergejsB, Smellyspice, Smithwesson, Snoozer, Still, TBS, Taduikis, Talo1019, Taylortech, Tgrand, Titan Wireless, Too, Tplecko, Uldis, Varstock, Ve2abc, Viktorc, Vitell, Weisiong84, Wkm001, Wpeople, Wrepinski, Wtechlink, Xeon, Zbuzanic

Bandwidth Managment and Queues *Source*: http://wiki.mikrotik.com/index.php?oldid=24769 *Contributors*: Aldalil, Alex rhys-hurn, Alexspils, Ashish, Awsmith, Djneoplan, Eising, Eugene, Fatonk, Fewi, Fisero, Fly man, JohnRBB, Jorgeamaral, Jorj, Maximan, MyThoughts, Nest, NetworkPro, Normis, Rock on all you f little dudes, SergejsB, Simply, Sudiptakp, Tony Burton, Valens

Firewall *Source*: http://wiki.mikrotik.com/index.php?oldid=25188 *Contributors*: Aashu, Alegara, Alexspils, Ashish, Butche, Chronos, Djneoplan, Eng Ma7mod, Epproach lyle, Eugene, FedeK, Fewi, Fox15rider, Gmsmstr, Herbison, Hrnous, Janisk, Jason@debian.org, Jp, Karlisb, Kostil, Lastguru, Letni, Marisb, Muhammad, NetworkPro, Normis, Qobtan, Reza.moghadam, Rieks, Savagedavid, SergejsB, Steveee, Stutteringp0et, Uldis, Vitell, Wg105, Wsun, Xinu

Monitoring *Source*: http://wiki.mikrotik.com/index.php?oldid=21708 *Contributors*: Alexspils, Ashish, Chupaka, Dsobin, Fewi, Judy213, Nest, Normis, Savage, SergejsB

User/Routing *Source*: http://wiki.mikrotik.com/index.php?oldid=24830 *Contributors*: 100mux, Aacable, Adi, Aegis, Andrewluck, Atis, Cdiggity, Chewbacca, Chronos, Cybercoder, Drunkers, Enk, Eugene, Fewi, HarvSki, Headstrong, Janisk, Karmasore, Kccoyote, Marek001, Marisb, Mariusol, Maximan, Mbeckwell, Megis, Miahac, Mneumark, Mplsguy, Normis, RK, Reza.moghadam, Route, Savagedavid, SergejsB, Tidar, Uldis

Scripts *Source*: http://wiki.mikrotik.com/index.php?oldid=25933 *Contributors*: Airstream, Alex rhys-hurn, Andreacoppini, Aruszek, Arve, Canniscam, Changeip, Cholegm, Chronos, Cmit, Dasiu, Datak, Davewilliamson, Davis, Dshove, Dsswiki, Dzintars, Earthstation, ElPablo, Elmauro, Enuro12, Eugene, Fbaffoni, Forne, GWISA, Giepie, Gregsowell, Gwicon, Hellbound, Herbison, Hjoelr, Illiniwireless, Infowest, Jason@debian.org, Jorgeamaral, Kostil, Krigevr, Lastguru, Ludwig, Mag, Marisb, Marko, Marks-mkt, Mattx86, Maxfava, Nahuelon, Nest, Normis, Ojsa, Omega-00, Paxy, Pedja, PoniTozheKoni, Rclark, Rick Frey, Riverron, Russ, Saik0, Savagedavid, Shmali, Skot, Steinmann, SurferTim, Tomaskir, Tplecko, Vitell, Wcsnet, Webasdf, Wpeople, Ziumus

Tunnels *Source*: http://wiki.mikrotik.com/index.php?oldid=25708 *Contributors*: Adjoneidi, Enk, Eugene, Fatonk, Hjoelr, Jesse.dupont, Mag, Marisb, Maximan, Mhammett, Miahac, MoySbh, Normis, Pingus, Reza.moghadam, Satman1w, Scrumi, SergejsB, Tplecko, Vladaz, Wispinternet

Wireless Setups *Source*: http://wiki.mikrotik.com/index.php?oldid=23078 *Contributors*: Andreacoppini, Andrisk, Bushy wiki, Dimitarv, Eugene, Johnalot, Makua, Maris, Marisb, Miki15, Mneumark, Mplsguy, Nest, NetworkPro, Normis, Omega-00, Reza.moghadam, Rick Frey, Route, SergejsB, Stuntshell, Sudiptakp, Tgrand, Valypetre

Manual:MPLS *Source*: http://wiki.mikrotik.com/index.php?oldid=23554 *Contributors*: Eising, Marisb, Mplsguy, Normis, Route, SergejsB

Manual:Virtualization *Source*: http://wiki.mikrotik.com/index.php?oldid=20250 *Contributors*: Danielillu, Janisk, Marisb, Normis

Use Metarouter to Implement Tor Anonymity Software *Source*: http://wiki.mikrotik.com/index.php?oldid=20602 *Contributors*: Webasdf

User/IPv6 *Source*: http://wiki.mikrotik.com/index.php?oldid=22353 *Contributors*: Jeeves, Marisb, Route, Tagno25

User Management *Source*: http://wiki.mikrotik.com/index.php?oldid=21600 *Contributors*: Cyph3r, Fewi, Janisk, Jp, Mudasir, Nest, Normis, Ntrits, SergejsB, SurferTim, Trevorlc1234, Vaello

The Dude *Source*: http://wiki.mikrotik.com/index.php?oldid=18423 *Contributors*: Adamd292, Bluecrow76, Bryanstein, Cajeptha, Dsobin, Dutchy, Eugene, GWISA, Huri, Lastguru, Lebowski, Mblanco, Mpegmaster, Nahuelon, Nest, Normis, Pikoro, Piwi3910, Rwilms, Sady, Savagedavid, Sdrenner, Uldis

User Manager *Source*: http://wiki.mikrotik.com/index.php?oldid=16431 *Contributors*: Akangage, Bhhenry, Binhtanngo2003, Cmit, Comnetisp, Eep, Girts, Hellbound, Janisk, Levipatick, Marisb, Nest, Normis, Polokus, Rtkrh10, SergejsB, Uldis

API PHP package *Source*: http://wiki.mikrotik.com/index.php?oldid=25927 *Contributors*: Marisb

API in C using winsock *Source*: http://wiki.mikrotik.com/index.php?oldid=23464 *Contributors*: Adenter, Janisk, Marisb

Manual:API Python3 *Source*: http://wiki.mikrotik.com/index.php?oldid=21594 *Contributors*: Janisk, Normis

API multiplatform from townet *Source*: http://wiki.mikrotik.com/index.php?oldid=21398 *Contributors*: Betti.enrico, Chronos

MikroNode *Source*: http://wiki.mikrotik.com/index.php?oldid=21397 *Contributors*: Chronos, Janisk, Trakkasure

API in Java *Source*: http://wiki.mikrotik.com/index.php?oldid=23596 *Contributors*: Chronos, Janisk

API In CPP *Source*: http://wiki.mikrotik.com/index.php?oldid=20848 *Contributors*: Creditrepairfixcredit, Marisb, Newacct, Valleyman86

Api php template *Source*: http://wiki.mikrotik.com/index.php?oldid=21396 *Contributors*: Chronos, Chupaka, Sw0rdf1sh

API in VB dot NET *Source*: http://wiki.mikrotik.com/index.php?oldid=19641 *Contributors*: Hex

RouterOS PHP class *Source*: http://wiki.mikrotik.com/index.php?oldid=17468 *Contributors*: Ataqlibert, Ayufan, Marisb

API command notes *Source*: http://wiki.mikrotik.com/index.php?oldid=25638 *Contributors*: Janisk, Ukasz

API Ruby class *Source*: http://wiki.mikrotik.com/index.php?oldid=20895 *Contributors*: Astounding, Janisk

Librouteros *Source*: http://wiki.mikrotik.com/index.php?oldid=14763 *Contributors*: Octo

API in C *Source*: http://wiki.mikrotik.com/index.php?oldid=23712 *Contributors*: Adenter, Nuclearcat, Webasdf

API ActionScript 3 class *Source*: http://wiki.mikrotik.com/index.php?oldid=12252 *Contributors*: Haakon

API Delphi Client *Source*: http://wiki.mikrotik.com/index.php?oldid=12245 *Contributors*: Chupaka

API Delphi *Source*: http://wiki.mikrotik.com/index.php?oldid=20580 *Contributors*: Eugenevdm, Normis, Rodolfo

API in C Sharp *Source*: http://wiki.mikrotik.com/index.php?oldid=13080 *Contributors*: C1982, Gregy, Normis

API PHP class *Source*: http://wiki.mikrotik.com/index.php?oldid=25769 *Contributors*: Blaze, Bmenking, Chronos, Cristiandeluxe, Denis Basta, Eugenevdm, Janisk, JeremyWJ, Mangia, Mmorier, Mpapec, Newacct, Normis, Piotr.piwonski, Tiagoratto, Viktorc, Vitell

MikroTik for Mac *Source*: http://wiki.mikrotik.com/index.php?oldid=22848 *Contributors*: Chenull, Donie, Eugene, Henkk78, Jeffsporos, Macsrwe, Myrrhman, Ni3ls, Normis, SergejsB, SomniusX, Tecpro, Tiukov, Zee

Assorted examples *Source*: http://wiki.mikrotik.com/index.php?oldid=23076 *Contributors*: Abakali, Ashish, Chronos, Chupaka, Dgerdes, Eugene, Fewi, Iron4umx, JJOliver998, Janisk, Jp, Laurinkus, Marisb, Maximan, Miahac, Nenad, Nest, Normis, Npero, Pedja, Prash in, Rieks, Sago-dan, Sergiom99, Shmali, Uldis, Webasdf

Image Sources, Licenses and Contributors

Image:Shot.jpg *Source*: http://wiki.mikrotik.com/index.php?title=File:Shot.jpg *License*: unknown *Contributors*: Normis

Image:Icon-note.png *Source*: http://wiki.mikrotik.com/index.php?title=File:Icon-note.png *License*: unknown *Contributors*: Marisb, Route

Image:Version.png *Source*: http://wiki.mikrotik.com/index.php?title=File:Version.png *License*: unknown *Contributors*: Normis

File:TorMikrotikDiagram.jpg *Source*: http://wiki.mikrotik.com/index.php?title=File:TorMikrotikDiagram.jpg *License*: unknown *Contributors*: Webasdf

File:Dude600.png *Source*: http://wiki.mikrotik.com/index.php?title=File:Dude600.png *License*: unknown *Contributors*: Normis

Image:api_studio_104.jpg *Source*: http://wiki.mikrotik.com/index.php?title=File:Api_studio_104.jpg *License*: unknown *Contributors*: Rodolfo

Image:Winbox_on_mac.jpg *Source*: http://wiki.mikrotik.com/index.php?title=File:Winbox_on_mac.jpg *License*: unknown *Contributors*: Ni3ls